



SiamNAS: Siamese Surrogate Model for Dominance Relation Prediction in Multi-objective Neural Architecture Search

Yuyang Zhou*

yuyang.zhou@nottingham.edu.cn

University of Nottingham Ningbo China
Ningbo, Zhejiang, China

Yew-Soon Ong

asyong@ntu.edu.sg

Nanyang Technological University
Centre for Frontier AI Research, Institute of High
Performance Computing, Agency for Science, Technology
and Research
Singapore, Singapore

Ferrante Neri*

f.neri@surrey.ac.uk

University of Surrey
Guildford, United Kingdom

Ruixin Bai†

ruixin.bai@nottingham.edu.cn

University of Nottingham Ningbo China
Ningbo, Zhejiang, China

Abstract

Modern neural architecture search (NAS) is inherently multi-objective, balancing trade-offs such as accuracy, parameter count, and computational cost. This complexity makes NAS computationally expensive and nearly impossible to solve without efficient approximations. To address this, we propose a novel surrogate modelling approach that leverages an ensemble of Siamese network blocks to predict dominance relationships between candidate architectures. Lightweight and easy to train, the surrogate achieves 92% accuracy and replaces the crowding distance calculation in the survivor selection strategy with a heuristic rule based on model size. Integrated into a framework termed SiamNAS, this design eliminates costly evaluations during the search process. Experiments on NAS-Bench-201 demonstrate the framework's ability to identify Pareto-optimal solutions with significantly reduced computational costs. The proposed SiamNAS identified a final non-dominated set containing the best architecture in NAS-Bench-201 for CIFAR-10 and the second-best for ImageNet, in terms of test error rate, within 0.01 GPU days. This proof-of-concept study highlights the potential of the proposed Siamese network surrogate model to generalise to multi-tasking optimisation, enabling simultaneous optimisation across tasks. Additionally, it offers opportunities to extend the approach for generating Sets of Pareto Sets (SOS), providing diverse Pareto-optimal solutions for heterogeneous task settings.

CCS Concepts

• Computing methodologies → Discrete space search.

*Both authors contributed equally to this research.

†Corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

GECCO '25, July 14–18, 2025, Malaga, Spain

© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 979-8-4007-1465-8/2025/07

<https://doi.org/10.1145/3712256.3726359>

Keywords

Neural Architecture Search, Surrogate Assisted Models, Siamese Networks, Multi-objective Optimisation

ACM Reference Format:

Yuyang Zhou, Ferrante Neri, Yew-Soon Ong, and Ruixin Bai. 2025. SiamNAS: Siamese Surrogate Model for Dominance Relation Prediction in Multi-objective Neural Architecture Search. In *Genetic and Evolutionary Computation Conference (GECCO '25), July 14–18, 2025, Malaga, Spain*. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3712256.3726359>

1 Introduction

Neural Architecture Search (NAS) automates the design and optimisation of neural networks, addressing the complex and resource-intensive task of model selection [53]. Traditionally reliant on human expertise and trial-and-error, this process is often slow and yields suboptimal results [23, 37]. This automation not only accelerates development but also enables the discovery of models that outperform manually designed counterparts, driving progress in domains such as computer vision and natural language processing.

Modern NAS frameworks extend beyond optimising model accuracy, recognising that practical applications require balancing multiple objectives. Key considerations include minimising parameter count, computational complexity, inference latency, and energy consumption [41]. These factors are particularly important for resource-constrained environments, such as smartphones and embedded systems, where efficiency is as critical as accuracy [27]. Multi-objective NAS frameworks address these challenges by employing advanced optimisation methods, such as Pareto-based strategies and evolutionary algorithms, to discover architectures that strike an optimal balance among competing requirements. This holistic approach ensures that the resulting models are not only high-performing but also efficient and scalable for diverse deployment scenarios.

NAS poses distinct challenges for generative models, requiring multi-objective optimisation beyond traditional approaches. As shown in [50], evaluating generative adversarial networks (GANs) involves not only minimising complexity but also optimising metrics such as Fréchet Inception Distance (FID) and Inception Score

(IS). This enables the design of efficient models that produce high-quality outputs.

NAS applications now extend well beyond image classification, covering areas such as medical diagnostics [2], driver assistance [40], and natural language processing [18]. These domains often involve highly complex and sometimes real-time constraints. As a result, enhancing computational efficiency through performance predictors—e.g., for accuracy, latency, and memory usage—has become crucial. At the same time, unified evaluation platforms like BenchENAS [46] are increasingly vital for algorithm development. Moreover, the growing demand for flexible architectures capable of addressing multiple tasks via techniques like transfer learning further compounds the challenge [8, 49].

All recent research directions of NAS as well as the challenges associated with them are characterised by their significant computational expense [34, 35, 55]. The inclusion of multiple objectives further exacerbates this computational cost. Specifically, multi-objective NAS requires the evaluation of a large number of solutions, which leads to the necessity of maintaining a large population in Evolutionary Multi-objective Optimisation (EMO). This added complexity makes it increasingly challenging to efficiently search for optimal architectures, as the number of function evaluations grows substantially with each additional objective [27]. This holds true for various types of neural architectures, as emphasised in [1].

For these reasons, a significant amount of effort in NAS research is dedicated to searching for new architectures while keeping the computational cost low, as reported in [25]. The strategies for reducing computational cost can be classified as follows:

- **Proxy-based NAS:** The computational cost is reduced by using shortcuts such as training on smaller datasets, training for fewer epochs, sharing the weights of a supernet, or evaluating scaled-down versions of architectures (e.g., [23, 51]).
- **Surrogate-assisted NAS:** The computational cost is reduced by using an auxiliary model, often a machine learning model, that predicts the performance of candidate architectures based on historical evaluation data. Rather than directly evaluating each architecture, these methods train a surrogate model to estimate performance, enabling rapid evaluation and prioritisation of promising candidates (e.g., [14, 44]).

Within the surrogate-assisted category, a prominent approach does not attempt to reconstruct the original expensive objective function directly. Instead, it transforms the objective into a task that aims to preserve accurate rankings among candidate architectures. For instance, the study in [20] employs a Graph Convolutional Network to predict the relative ranking of architectures within an evolutionary cycle. Similarly, the works in [42, 47] develop models to predict pairwise rankings, facilitating the selection process within NAS. Following the same logic, the study in [16] proposes leveraging isomorphism to inexpensively generate additional training data for graph-based encoding. Furthermore, an isomorphic score is introduced to predict the performance of candidate architectures.

The use of surrogate models in multi-objective NAS is inherently more complex than in single-objective cases, as it involves managing a greater number of interdependent components. These components can be computationally expensive, necessitating the

use of strategies to mitigate these costs. The study in [28] formulates NAS as a bi-level optimisation problem, employing surrogate models at both levels. At the upper level (architecture optimisation), four surrogate models are trained iteratively, and an adaptive mechanism selects the most accurate model for performance prediction. For the lower level (weights optimisation), a pre-trained supernet provides warm-start weights to candidate architectures, reducing the time required for evaluation. For real-time semantic segmentation, NAS is treated as a multi-objective problem in [26], using online and offline surrogate models to predict accuracy and latency, with a hierarchical prescreening strategy to balance both. [29] employs cross-task transfer learning with iterative source selection to avoid negative transfer. [30] proposes an online classifier for dominance prediction, supported by adaptive clustering and α -domination to address class imbalance.

This paper, inspired by the ranking logic in single-objective NAS proposed in [16] and the dominance predictor introduced in [29], presents a novel surrogate model designed to predict dominance between pairs of architectures within multi-objective NAS frameworks. The proposed model is a modified version of a Siamese network, utilizing its few-shot learning capability. This approach leverages the fact that the surrogate can be efficiently trained with a limited number of samples.

The proposed surrogate model does not aim to approximate any objective function. Instead, it takes two input architectures and predicts the dominance relationship between them. This approach is proposed here because an accurate estimation of the dominance relationship between pairs of solutions is what is truly needed to map a population into non-dominated sets, making the approximation of an objective function unnecessary. Furthermore, a comparator-like surrogate model requires simpler architecture than an approximator and can be trained with a modest amount of data.

Similar to a Siamese network, the model consists of two identical sub-networks—here implemented as a simple Multi-Layer Perceptron (MLP) network. However, unlike a traditional Siamese network, the proposed model has three possible outcomes: dominance of the first architecture, dominance of the second architecture, or non-dominance between the two architectures. The surrogate can be efficiently trained using a few samples from the search space and seamlessly integrated into the non-dominated sorting mechanisms of many popular multi-objective optimisers.

The key contributions of this paper are as follows:

- A **novel surrogate model**, inspired by Siamese networks and leveraging **few-shot learning**, by training the surrogate model by a few ground truth performance metrics, is introduced for predicting the **dominance relationship** between pairs of architectures in multi-objective NAS.
- The proposed surrogate model extends the standard Siamese network architecture by incorporating **three potential outputs** instead of two. These outputs represent the **three possible outcomes** of a dominance comparison: dominance of the first architecture, dominance of the second architecture, or non-dominance. This dominance prediction is made by **combining the outputs** of multiple Siamese surrogate blocks in an **ensemble model**, with the final prediction determined by two subsequent majority votes.

- A novel **multi-objective NAS framework** based on the **Non-dominated Sorting Genetic Algorithm II (NSGA-II)** leveraging this surrogate model inspired by Siamese networks is proposed. The framework, namely SiamNAS, optimises **three objectives**: model accuracy, the number of parameters, and the number of floating-point operations (FLOPs) required to compute the output from a given input. This framework incorporates a modified survivor selection mechanism designed to enable efficient selection with minimal computational cost.

The remainder of this paper is structured as follows. Section 2 introduces the notation, problem formulation, and a brief overview of Siamese networks. Section 3 describes the architecture and components of the proposed surrogate model. Section 4 presents the SiamNAS framework and its integration with the surrogate. Section 5 reports experimental results, and Section 6 concludes the paper.

2 Problem Formulation and Background

If we denote a vector containing the architectural parameters as \mathbf{x} , identifying an architecture, Ω the set containing all the possible architectures, and \mathbf{w} the vector of weights associated with a candidate architecture \mathbf{x} , NAS can be formulated as the following multi-objective optimisation problem [27].

$$\min_{\mathbf{x}} (\mathbf{f}_e(\mathbf{x}; \mathbf{w}^*), \mathbf{f}_c(\mathbf{x}), \mathbf{f}_H(\mathbf{x})) \quad (1)$$

$$\text{s.t. } \mathbf{w}^* \in \operatorname{argmin}_{\mathbf{w}} \mathcal{L}_{trn}(\mathbf{x}, \mathbf{w}), \mathbf{x} \in \Omega$$

where \mathcal{L}_{trn} is the loss function used to train the architecture, \mathbf{w}^* indicates the weights of the candidate architecture \mathbf{x} after training, \mathbf{f}_e is the vector of functions associated with the performance of the network, e.g. through its prediction error, \mathbf{f}_c is the vector of functions associated with the complexity of the network (or more generally the model), e.g. the energy consumption of the model, \mathbf{f}_H is the data structure associated with hardware performance.

In this article, we perform the search of the architecture by optimising three objective functions, one of type \mathbf{f}_e and two of type \mathbf{f}_c :

$$\mathbf{f}_e(\mathbf{x}) = 1 - acc(\mathbf{x}, D) \quad (2)$$

$$\mathbf{f}_c(\mathbf{x}) = (\#params(\mathbf{x}), FLOPS(\mathbf{x})) \quad (3)$$

where, $acc(\mathbf{x}, D)$ denotes the accuracy of the architecture \mathbf{x} trained on the target dataset D , $\#params(\mathbf{x})$ represents the number of parameters in \mathbf{x} , and $FLOPS(\mathbf{x})$ refers to the number of floating-point operations required by \mathbf{x} to calculate an output for a given input. It is noteworthy that, as documented in [30], $\#params(\mathbf{x})$ and $FLOPS(\mathbf{x})$ constitute conflicting objectives.

2.1 Siamese Networks

Siamese neural networks are a class of neural architectures designed to learn meaningful representations by comparing pairs of inputs. Introduced by Bromley et al. (1994) for signature verification tasks, these networks consist of two or more identical subnetworks that share weights and operate on different inputs in parallel. The shared architecture ensures that both inputs are transformed into comparable feature representations, making them particularly effective for tasks like similarity learning and metric-based decision-making. The output of a Siamese network is typically a similarity score,

computed through a distance metric (e.g., Euclidean or cosine distance) applied to the learned representations. Variants of Siamese networks have since been adapted for various domains, including facial recognition [4] and few-shot learning.

3 Siamese Network Surrogate Model

In this work, we propose a novel modification to the Siamese neural network architecture to serve as a surrogate model for predicting the dominance relationship in multi-objective problems between two candidate architectures. Conceptually, the proposed model consists of an ensemble of Siamese networks. Each Siamese network is composed of two identical multilayer perceptron (MLP) networks, which process input vectors representing a pair of architectures, denoted as \mathbf{x}_1 and \mathbf{x}_2 . The proposed surrogate model evaluates \mathbf{x}_1 and \mathbf{x}_2 to determine which architecture, if any, dominates the other. Each Siamese network is trained separately and thus has its own set of weights. Within each Siamese network that composes the model, the weights of the two MLP networks are shared.

While Siamese networks were originally designed for tasks such as comparing facial features or fingerprints, we adapt this concept to the domain of dominance evaluation. The proposed surrogate model accommodates three possible outcomes: $\mathbf{x}_1 \preceq \mathbf{x}_2$, $\mathbf{x}_2 \preceq \mathbf{x}_1$, or non-dominated solutions.

3.1 Encoding

As mentioned above, the input architectures are represented as encoded vectors, \mathbf{x}_1 and \mathbf{x}_2 . To demonstrate the effectiveness of the proposed approach, this study utilizes the NAS-Bench-201 search space [11], where each candidate architecture is a convolutional neural network (CNN) block with six potential connections between four layers. Each connection can be assigned one of five possible operators, and the original representation as a 6-dimension vector specifies both the presence of connections and their associated operators. This representation is converted into a one-hot encoding format, consisting of a 6×5 matrix. In this matrix, each row corresponds to a potential connection, and each column represents a specific operator. For each connection, a single bit in the row is set to 1, indicating the active operator, while all other bits are set to 0. Thus, each candidate architecture is represented as a binary vector of length 30. An illustration of the encoding of a NAS-Bench-201 into a vector \mathbf{x} is provided in Fig. 1.

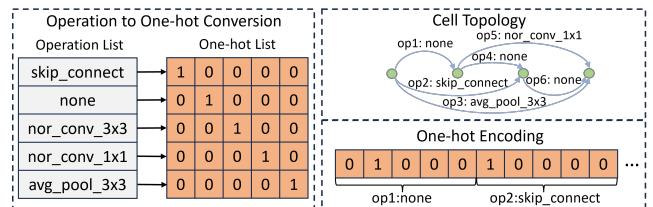


Figure 1: Encoding a network from the NAS-Bench-201 search space as a binary vector \mathbf{x} to serve as input for the Siamese surrogate predictor.

One-hot encoding is employed because it provides a clear, unbiased, and machine-readable format for categorical data [38]. This

encoding avoids introducing any ordinal relationships between operators, ensuring that each operator is treated independently. Furthermore, it is compatible with the neural network architecture used in this study, facilitating effective processing and comparison of candidate architectures. By standardizing the input format, one-hot encoding ensures an efficient and semantically meaningful representation tailored to the task at hand.

3.2 Multi-Layer Perceptron Networks within the Surrogate Model

As mentioned in the general description of the surrogate model, the proposed surrogate model consists of two identical MLP networks whose weights are shared between each other. Each MLP comprises one hidden layer, which is a fully connected layer with 32 neurons, followed by the ReLU activation function. For each MLP, the binary input vectors x_1 and x_2 are processed independently. At the hidden layer, these inputs are transformed into real-valued embedding vectors, b_1 and b_2 , each of length 32. These embeddings are then combined by computing their difference, resulting in the vector $d = b_1 - b_2$. Naturally, in the case of identical architectures, i.e., $x_1 = x_2$, it follows that $d = \mathbf{0}$, where $\mathbf{0}$ denotes the null vector.

The difference vector d is processed by another MLP, which consists of two fully connected layers. The first layer comprises 32 neurons (processing d) followed by a ReLU activation function, while the second layer contains a single neuron that serves as the output. This output layer utilizes a sigmoid activation function, ensuring that the output is restricted to the range $(0, 1)$.

The two MLP networks share identical weights and are trained using the Adam optimiser to minimize the binary cross-entropy loss [17]. The same optimiser and loss function are also employed to train the final part of the network, indicated as MLP Classifier in the yellow square depicted in Fig. 2.

3.3 Ensemble-based Dominance Prediction

The output, as described in Section 3.2, is a scalar in range $(0, 1)$ which is rounded to $R \in \{0, 1\}$. It represents the dominance relationship of x_1 and x_2 . Specifically, $R = 1$ indicates that $x_1 \preceq x_2$, i.e. x_1 dominates x_2 , while $R = 0$ implies that $x_1 \preceq x_2$ with probability 0, i.e., $x_1 \not\preceq x_2$. In the case of identical architectures, $d = \mathbf{0}$ and $R = 0$, which verifies the condition $x_1 \not\preceq x_2$. Finally, the rounded value R serves as the outcome of surrogate model prediction.

In the proposed implementation, to enhance the performance of the model, multiple Siamese surrogate blocks, each composed of two identical MLP networks and a separate MLP classifier to process the embedding vector and calculate the output R , are trained independently. These surrogate models are then combined in an ensemble approach to assess the dominance relationship between the two architectures. Specifically, x_1 and x_2 are fed into each surrogate model in the ensemble, generating predictions for R . Let N_m denote the number of surrogate models, yielding predictions R_1, R_2, \dots, R_{N_m} . After rounding predictions to either 0 or 1, each surrogate model can be viewed as a classifier. If the majority of these classifiers predict an outcome of 1, then it is predicted that $x_1 \preceq x_2$. Otherwise, the order of the inputs (i.e., the candidate architectures x_1, x_2) is inverted, and the outputs are recalculated by the surrogate models to predict whether $x_2 \preceq x_1$. If the two

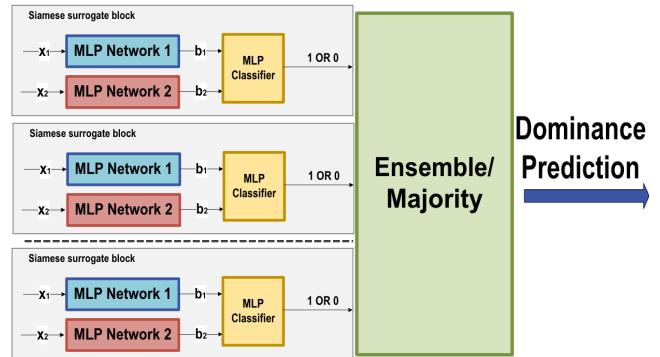


Figure 2: Implementation of the Siamese network surrogate model: An ensemble of multiple Siamese network blocks is utilised to predict dominance relationships, with final predictions determined through a majority voting mechanism.

consecutive predictions are $x_1 \not\preceq x_2$ followed by $x_2 \not\preceq x_1$, the model concludes that the two architectures do not dominate each other. An illustration of the implemented scheme is shown in Fig. 2.

4 SiamNAS: A Multi-objective Algorithm for Neural Architecture Search

The Siamese network surrogate model, described in Section 3, is then integrated into the multi-objective NAS framework, here indicated as SiamNAS and presented in the following sections.

4.1 Overall Framework

The pseudocode for the overall framework of the proposed SiamNAS is provided in Algorithm 1. The framework is divided into three phases: Phase 1 (lines 1-2) focuses on building the ensemble of Siamese network surrogate models, see Section 4.2; Phase 2 (lines 3-12) performs evolutionary NAS; and Phase 3 (line 13) obtains the final promising Pareto front from the last population.

It is important to note that in Phases 1 and 3, the true performance metrics of the candidate architectures are calculated, in this case queried from the NAS-Bench-201. In Phase 1, the number of architecture evaluations through true objective function calculations is equal to the number of candidate architectures selected for training, N_s . In Phase 3, the number of queries corresponds to the size of the promising Pareto front in the last population, \mathcal{P} , which is not greater than the population size N . In total, since no true objective function calculations are performed in Phase 2, at most $N_s + N$ architecture evaluations are made using true objective function calculations.

In Phase 2 (lines 3-12), evolutionary operations are performed on a population of N architectures within a multi-objective evolutionary framework. At each generation, solutions are selected for a mating pool using binary tournament selection, where pairwise comparisons are made with the Siamese surrogate model described in Section 3. The selected solutions undergo uniform crossover and flip mutation with rates R_c and R_m , respectively. A check is performed on the newly generated offspring architectures to ensure their validity and meaningfulness. In this case, the search space S

Algorithm 1 SiamNAS Overall Framework

INPUT: \mathcal{S} : search space , N : population size , T : number of generations, R_c : crossover rate, R_m : mutation rate, M : Siamese network surrogate model, N_s : number of samples to train Siamese network base surrogate ensemble, N_m : number of Siamese surrogate models

OUTPUT: \mathcal{P} : A set of solutions on the same Pareto front

- 1: $D_s \leftarrow$ randomly sample N_s architectures from the search space \mathcal{S} and calculate the three true objective function values for \mathbf{f}_c as per eq. (2) and (3) # Phase1
- 2: $M \leftarrow$ Use these N_s architectures to train an ensemble of N_m Siamese surrogate models, see Algorithm 2
- 3: $P \leftarrow$ initialize a population of size N from \mathcal{S} # Phase2
- 4: $t \leftarrow 0$
- 5: **while** $t < T$ **do**
- 6: $P' \leftarrow$ binary tournament selection by M from P
- 7: $P' \leftarrow$ apply uniform crossover to the architectures in P' with crossover rate R_c
- 8: $P' \leftarrow$ apply flip mutation to each bit of the architectures in P' with mutation rate R_m
- 9: $P' \leftarrow$ check that all the architectures in P' are meaningful
- 10: $P \leftarrow$ Biased-Selection(N, M, P, P') by Algorithm 3
- 11: $t \leftarrow t + 1$
- 12: **end while**
- 13: $\mathcal{P} \leftarrow$ select the final population from P and calculate the true objective function values # Phase3

corresponds to that of the NAS-Bench-201 benchmark [11]. Architectures that do not conform to the expected format are repaired by ensuring that each group of five bits, which is the one-hot representation of operations, contains exactly one bit set to 1, as illustrated in Fig. 1. If multiple bits are set to 1 within a group, one is randomly chosen to remain 1, while the others are forced to 0. Conversely, if no bits are set to 1, a bit is randomly selected and flipped from 0 to 1. The survivor selection is then applied by means of a novel operator based on the proposed Siamese surrogate model. The details of this novel operator, namely Biased Selection, is outlined in Section 4.3.

The scheme illustrating the functionality of the SiamNAS framework is presented in Fig. 3.

4.2 Data Management and Training of the Siamese Network Surrogate Model

As detailed in Phase 1 (lines 1–2) of Algorithm 1, N_s candidate architectures are sampled from the search space, and their true objective function values are calculated. These architectures form a dataset to train the Siamese surrogate model. However, since the proposed surrogate model requires two inputs, pairs of inputs must be constructed. Consequently, from these N_s candidate architectures evaluated through the true objective functions, $N_s \times N_s$ possible training entries can be generated. This approach significantly enlarges the available data for training without requiring additional evaluations of true objective functions.

With this dataset of $N_s \times N_s$ entries, the Siamese surrogate model is trained by following Algorithm 2. This algorithm is divided into

three main stages: pair construction, pair reassignment, and training. In the first stage (lines 5–6), the N_s candidate architectures comprising D_s are shuffled to generate a shuffled dataset D_p . The generic elements D_s^i and D_p^j are then paired to create a dataset of pairs, D'_t , as shown in line 4.

During the creation of pairs, it is crucial to address the issue of class label imbalance, which is particularly important for surrogate models predicting dominance in multi-objective optimisation. As highlighted in [30], ensuring that positive class, the cases where $\mathbf{x}_1 \preceq \mathbf{x}_2$ (scoring 1), are roughly balanced with negative class, the cases where $\mathbf{x}_1 \not\preceq \mathbf{x}_2$ (scoring 0), is essential for accurate training of the surrogate model. It is worth noting that, even with just three objectives, a simple random sampling of pairs is likely to result in a dataset dominated by negative class, exacerbating the class imbalance problem. To mitigate the issue of class imbalance, the constructed pairs are reassigned to reduce the proportion of negative labels (label 0) to approximately 50% (lines 7–14). A resampling rate, θ , is calculated according to the proportion of negative pairs in the dataset (lines 7–8). Each negative pair is then given a chance, with probability θ , to have its second candidate replaced. This replacement continues until a candidate is found that is dominated by the first candidate in the original pair (lines 9–14). This reassignment process ensures a more balanced dataset, which is essential for the effective training of the surrogate model.

After balancing the proportion of negative and positive pairs, labels l are assigned to all pairs. If the first candidate in a pair dominates the second candidate, a label of 1 is assigned; otherwise, a label of 0 is assigned. These pairs of candidates, D_t , together with their corresponding labels, l , are used to train one Siamese surrogate block $m \in \{m_1, m_2, \dots\}$ in the ensemble. Each Siamese surrogate block is trained by different pairs of candidates D_t , constructed by the same true objective function values D_s . Once trained, these blocks collectively form the Siamese Network Surrogate Model, M .

4.3 Biased Selection Criterion

With reference to line 10 of Algorithm 1, a modified survivor selection strategy that leverages the trained surrogate model is proposed. First, the population of parents, P , is merged with the offspring population, P' , resulting in an auxiliary population, Q . This population, Q , is then divided into non-dominated sets (fronts) by applying the efficient non-dominated sorting method described in [52], where each comparison is performed using the proposed Siamese network surrogate model (lines 1–2 of Algorithm 3).

In the classical multi-objective optimisation algorithm NSGA-II [7], the crowding distance is employed as a secondary criterion, following non-dominated sorting, to decide which candidates should be retained for the next generation. However, in our case, as the true objective function values are not available, the calculation of the crowding distance is infeasible without resorting to true objective function evaluations, which would negate the computational efficiency achieved thus far.

To address the absence of crowding distance in NSGA-II, we propose a simple heuristic method detailed in Algorithm 3 that slightly amends the popular NSGA-II selection mechanism outlined in [7]. In this study, instead of retaining candidates with the largest crowding distance, we prioritise the selected candidates with the

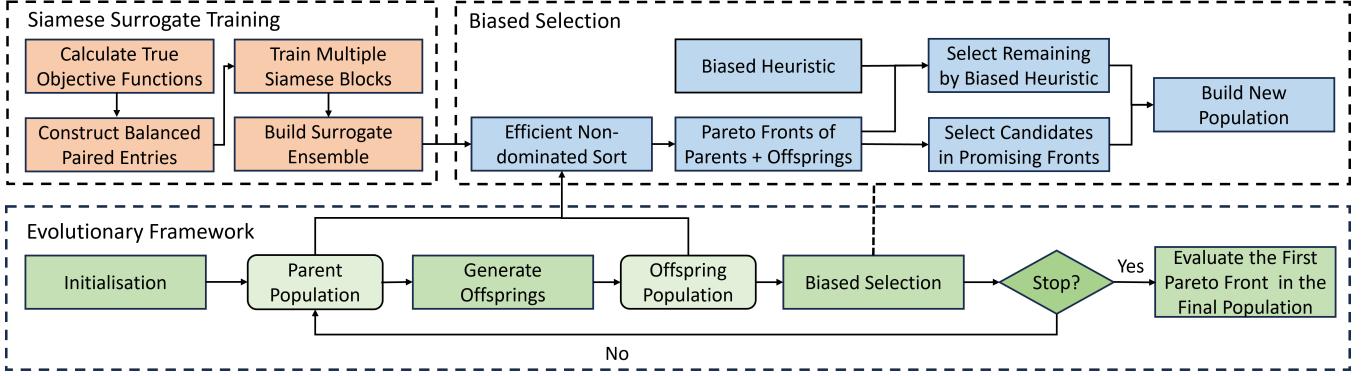


Figure 3: Framework of SiamNAS: Green blocks represent the evolutionary operators, where offspring generation and biased selection leverage the surrogate model for dominance-based comparisons. Operations related to training the surrogate model are highlighted in orange, while the biased selection process is marked in blue.

Algorithm 2 Train-Surrogate-Ensemble

INPUT: M : Siamese network base surrogate ensemble, D_s : samples from search space for training, N_m : number of Siamese surrogate models
OUTPUT: M : trained Siamese Network Surrogate Model (based on the ensemble of Siamese network blocks)

```

1: for  $r = 1 \dots N_m$  do
2:    $m_r \leftarrow$  initialize Siamese surrogate model
3:    $D_t \leftarrow \emptyset$ 
4:   for  $i = 1 \dots 100$  do
5:      $D_p \leftarrow$  randomly shuffle  $D_s$ 
6:      $D'_t = \{(D_s^1, D_p^1), \dots, (D_s^{N_s}, D_p^{N_s})\} \leftarrow$  combine each candidate in  $D_s$  and  $D_p$  to build pairs of data
7:      $\sigma \leftarrow$  proportion of pairs in  $D'_t$  where  $D_s^*$  dominates  $D_p^*$ 
8:      $\theta \leftarrow 0.5/(1 - \sigma)$ 
9:     for  $j = 1, \dots, N_s$  do
10:        $r \in \mathbb{R} \leftarrow$  uniformly distributed random value in  $[0, 1]$ 
11:       if  $D_s^j$  does not dominate  $D_p^j$  and  $r < \theta$  then
12:          $D_p^j \leftarrow$  find a  $D_p^k \in D_s$  that  $D_s^j$  dominates  $D_p^k$  if exists
13:       end if
14:     end for
15:      $D_t \leftarrow D_t \cup D'_t$ 
16:   end for
17:    $l \leftarrow$  assign label to each pair in  $D_t$ , where the label is 1 if the first candidate in pair dominates the other, otherwise 0
18:    $m_r \leftarrow$  train Siamese surrogate model by  $D_t$  and  $l$ 
19: end for
20:  $M \leftarrow$  build ensemble by trained Siamese surrogate models
 $m_1 \dots m_R$ 

```

largest number of parameters during the survivor selection phase (lines 8–12). The rationale behind this choice is twofold:

- (1) The number of parameters is a training-free metric. Acquiring this information incurs significantly lower computational cost compared to evaluating the true objective functions such as accuracy.

- (2) Empirical evidence suggests that larger neural network models, characterised by a greater number of parameters, often exhibit superior performance.

In other words, the preference for larger models is employed as a proxy for performance, serving to counterbalance the minimisation of the number of parameters. Conversely, favouring smaller models within the non-dominated set would bias the search towards architectures likely to exhibit unsatisfactory performance.

Algorithm 3 Biased-Selection

INPUT: N : population size, M : Siamese network base surrogate ensemble, P : parent population, P' : offspring population
OUTPUT: Q : New population

```

1:  $Q' \leftarrow P \cup P'$ 
2:  $\mathcal{F} = \{F_1, F_2, \dots, F_k\} \leftarrow$  sequential efficient non-dominated sort  $Q'$  by  $M$ 
3:  $Q \leftarrow \emptyset, i = 1$ 
4: while  $|Q| + |F_i| \leq N$  and  $i \leq k$  do
5:    $Q \leftarrow Q \cup F_i$ 
6:    $i \leftarrow i + 1$ 
7: end while
8: if  $|Q| < N$  then
9:    $N_r \leftarrow N - |Q|$ 
10:   $F'_i = \{s_1, s_2, \dots\} \leftarrow$  sort  $F_i$  in descending order by the number of trainable parameters of the candidate neural architecture
11:   $Q \leftarrow Q \cup \{s_1, \dots, s_{N_r}\}$ 
12: end if

```

5 Experimental Results

The proposed SiamNAS framework was evaluated on the NAS-Bench-201 [11] search space using three datasets: CIFAR10 [19], CIFAR100 [19], and ImageNet-16-120 [39]. To train the Siamese network surrogate model, a sample of $N_s = 600$ architectures with true objective function values, namely the train accuracy, the number of parameters and the FLOPs, was selected. Training was performed using the ADAM optimiser [17] over 20 epochs, with a learning

Table 1: Ablation study of the impact of ensemble size N_m on the prediction accuracy and F1 score on of the Siamese network surrogate model over NAS-Bench-201 and for CIFAR10, CIFAR100, and ImageNet-16-120.

N_m	CIFAR10		CIFAR100		ImageNet-16-120	
	Accuracy	F1 Score	Accuracy	F1 Score	Accuracy	F1 Score
1	90.59	0.70	90.78	0.70	90.40	0.73
3	92.00	0.74	91.04	0.71	91.74	0.76
5	92.64	0.76	91.68	0.72	92.06	0.77
7	92.83	0.76	91.61	0.72	92.57	0.79
9	93.02	0.77	91.36	0.71	92.57	0.79
11	92.77	0.76	91.42	0.72	92.38	0.78
13	92.83	0.76	91.49	0.72	92.45	0.78

rate of 0.001, a batch size of 100, and binary cross-entropy as the loss function. The ensemble consisted of seven Siamese network blocks.

Within the evolutionary framework, the population size N was set to 50, the crossover rate R_c to 0.7, the mutation rate to 0.1, and the total number of generations T to 2000. Each experiment was repeated 10 times to ensure robustness, with all computations performed on the same machine equipped with a single NVIDIA GeForce RTX 4090 GPU card.

5.1 Ablation Studies

The design of SiamNAS and its hyperparameter settings were determined through comprehensive ablation studies. In this section, we examine the impact of the ensemble size, N_m , on accuracy. The experimental settings of the ablation study follow the same as those specified in Section 5, while the number of Siamese blocks in one ensemble is altered from 1 to 13. The results, presented in Table 1, indicate that using seven Siamese network blocks yields at least the second best results in terms of both accuracy and F1 score. Furthermore, increasing the ensemble size beyond this point does not result in any significant improvement.

Furthermore, we present the impact of the training dataset size, N_s , on the accuracy of the predictive Siamese network surrogate model. While all other experimental settings are the same as introduced at the beginning of Section 5, the surrogate model was tested with N_s ranging from 0 to 1000, and the results are summarised in Table 2. The findings reveal a trade-off between the model accuracy/F1 and the number of architectures for which true objective function values are calculated. Additionally, the corresponding GPU computational costs are provided. Notably, for $N_s = 600$, the surrogate model achieves an accuracy of approximately 92%, with further increases in N_s offering no significant improvement in predictive accuracy.

5.2 Results on NAS-Bench-201

The performance of the proposed SiamNAS on NAS-Bench-201 was evaluated across CIFAR-10, CIFAR-100, and ImageNet-16-120 datasets and compared against twenty-six state-of-the-art algorithms from the literature. Among these competitors, eight algorithms search within the NAS-Bench-201 space without surrogate models (upper part of Table 3), employing alternative mechanisms such as weight inheritance to reduce computational cost. The remaining eighteen algorithms incorporate surrogate models in their

Table 2: Comparison of surrogate models prediction performance with different number of true objective function calculations, i.e., N_s on NAS-Bench-201.

N_s	Cost (GPU Days)	CIFAR10		CIFAR100		ImageNet-16-120	
		Accuracy	F1 Score	Accuracy	F1 Score	Accuracy	F1 Score
None	N/A	15.56	0.24	11.65	0.21	14.21	0.25
100	1.67e-3	86.36	0.48	86.43	0.53	88.54	0.62
200	1.99e-3	88.28	0.60	90.14	0.62	89.44	0.62
400	2.77e-3	90.91	0.71	91.68	0.69	91.04	0.71
600	3.65e-3	92.45	0.71	93.41	0.77	91.87	0.73
800	4.56e-3	92.19	0.75	93.09	0.75	92.70	0.80
1000	5.63e-3	92.70	0.78	92.57	0.74	92.57	0.77

search process (lower part of Table 3). The detailed comparison results are presented in Table 3 where Oracle indicates the performance of the best architecture searchable in the NAS-Bench-201 space.

Experimental results on the accuracy of the searched models demonstrate that SiamNAS achieves excellent performance on CIFAR-10, successfully identifying the theoretical optimum and outperforming all competitors. Additionally, SiamNAS exhibits competitive performance on CIFAR-100 and ImageNet-16-120. To further validate its effectiveness, the Siamese network surrogate models trained on CIFAR-10 was applied to the classification tasks on the CIFAR-100 and ImageNet-16-120 datasets. The experimental results indicate evidence of positive transfer, with the architecture achieving very good performance on both datasets. Interestingly, the results obtained through transfer learning slightly surpass those achieved when using the datasets directly, where the ImageNet-16-120 error rate is the second best among all methods.

It is worthwhile to highlight several key points. First, SiamNAS is a multi-objective NAS algorithm, whereas most of the compared algorithms are single-objective. In this context, the performance of SiamNAS is even more impressive. Second, the number of architectures evaluated using true objective functions is 600 for SiamNAS, compared to as few as 100 for other surrogate-based models (and as low as 90 in the case of ReNAS [48]). This discrepancy arises because the sorting process in multi-objective optimization is inherently more complex than in the single-objective case. Lastly, the algorithm runtime (excluding the time to perform the evaluations of architectures through true objective functions) of SiamNAS is significantly lower than that of all its competitors. For instance, SiamNAS has a runtime of roughly 0.01 GPU days, whereas BANANAS [45] requires 2 GPU days as shown in Table 4.

An example of the final non-dominated set identified by SiamNAS for the CIFAR10 dataset is illustrated in Fig. 4. The architecture with the highest accuracy is observed to have a significantly large number of parameters (measured in megabytes) and an exceptionally high number of FLOPs. Specifically, this architecture is characterized by #params = 1.074 MB , FLOPs = $1.533 \times 10^2 \text{ M}$, and $1 - \text{acc} = 5.627$. Conversely, the architectures with minimal FLOPs and parameter counts, which are clustered closely in this example, exhibit suboptimal performance. These architectures are defined by #params = $7.331 \times 10^{-2} \text{ MB}$, FLOPs = 7.783 M , and $1 - \text{acc} = 3.078 \times 10^1$. A solution with a balanced trade-off among the three objectives is marked in blue and associated with the objective function values #params = $3.444 \times 10^{-1} \text{ MB}$, FLOPs = $4.711 \times 10^1 \text{ M}$, and $1 - \text{acc} = 6.837$.

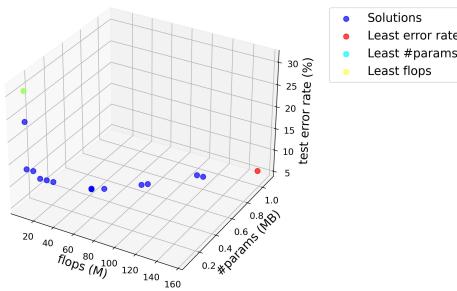


Figure 4: The solutions in a Pareto front found by SiamNAS on CIFAR10 dataset on NAS-Bench-201.

Table 3: Comparison of NAS algorithms on NAS-Bench-201.

NAS Algorithm	CIFAR10	CIFAR100	ImageNet-16-120
	Test Error Rate	Test Error Rate	Test Error Rate
RSPS [21]	8.95±0.66	31.74±0.96	59.31±0.36
SETN [9]	7.28±0.73	30.64±1.72	60.49±0.33
ENAS [33]	6.24±0.00	29.33±0.62	58.56±0.00
FairNAS [6]	6.77±0.18	29.00±1.46	57.81±0.31
DARTS [5]	6.20±0.40	28.47±1.51	54.88±0.82
BOHB [12]	6.06±0.28	28.00±0.86	54.30±0.86
REINFORCE [54]	6.10±0.26	28.14±0.89	54.36±0.78
GDAS [10]	6.77±0.58	31.83±2.50	60.60±0.00
NAR [15]	5.67±0.05	27.11±0.37	53.34±0.23
GenNAS-N [22]	5.82±0.10	27.44±0.74	54.41±0.54
NASWOT [31]	7.04±0.81	30.02±1.22	55.56±2.10
MbML-NAS (RF) [32]	6.64±0.20	29.67±0.85	57.01±4.21
MbML-NAS (GB) [32]	6.97±0.52	29.98±1.17	55.72±1.42
NASWOT [31]	7.19±0.99	30.52±1.70	56.90±3.16
Random Search [13]	6.18±0.24	28.60±0.83	54.64±0.67
REA [36]	5.94±0.29	27.28±0.72	54.01±0.51
BANANAS [45]	5.77±0.30	26.75±0.63	53.69±0.31
GP bayesopt [45]	5.84±0.31	26.95±0.75	53.75±0.34
DNGO [13]	5.92±0.26	27.34±0.67	54.00±0.48
Bohamiann [45]	5.91±0.26	27.35±0.66	53.97±0.48
GCN Predictor [45]	6.26±0.33	28.57±0.77	54.64±0.78
BONAS [13]	5.68±0.15	26.70±0.46	53.69±0.30
NPENAS-SSRL [43]	5.68±0.19	26.53±0.30	54.17±0.60
NPENAS-CCL [43]	5.68±0.19	26.51±0.23	54.39±0.41
SAENAS-NE [13]	5.66±0.12	26.54±0.20	53.64±0.26
ReNAS [48]	6.01±0.25	27.88±0.79	54.03±0.49
SiamNAS(ours)	5.63±0.00	27.39±0.31	54.07±0.08
SiamNAS-transfer(ours)	5.63±0.00	27.03±0.19	53.63±0.12
Oracle	5.63	26.49	52.69

Table 4: Comparison of NAS algorithm runtime on NAS-Bench-201.

NAS Algorithm	Time Cost (GPU Days)	Method
DARTS [5]	4	Gradient
BANANAS [45]	2	BO + neural predictor
NAR [15]	0.002	Sampling + neural predictor
ENAS [33]	0.15	Gradient
SETN [9]	0.35	Gradient + neural predictor
ReNAS [48]	0.001	EA + neural predictor
SiamNAS(ours)	0.01	EA + neural predictor

6 Conclusion and Future Work

This article presents a novel surrogate model based on an ensemble of Siamese network blocks, each trained to predict the outcome of

pairwise comparisons between candidate architectures. This approach is applied within the context of multi-objective NAS, aiming to identify accurate models with limited parameters and computational cost. Designed to be efficient and straightforward to train, the current surrogate model achieves an accuracy of approximately 92% on predicting the outcome of pairwise comparisons. To demonstrate its potential, the model was integrated into the non-dominated sorting process of a multi-objective NAS framework, which seamlessly replaces the crowding distance calculation with a simple heuristic rule based on candidate architecture size. This integration allows SiamNAS to rely solely on the surrogate model, avoiding the need to train models during the search process.

The results from this proof-of-concept study on NAS-Bench-201, using three datasets, highlight the effectiveness of the proposed algorithm in achieving excellent performance while significantly reducing computational costs. As this work primarily serves as a foundation, future research will involve testing the proposed framework on real-world architectures beyond the constraints of benchmark datasets, further validating its applicability and robustness.

The Siamese network surrogate model introduced in this study holds promising potential for extending its application to multi-tasking optimisation [3], particularly in the context of generating Sets of Pareto Sets (SOS) [24]. In SOS problems, where the objective is to create task-specific Pareto-optimal sets across heterogeneous environments, efficient methods for predicting dominance relationships are critical. A natural extension of this work could involve adapting the surrogate model to multi-tasking scenarios by leveraging its ability to learn and generalise dominance patterns across diverse tasks. This could facilitate the identification of commonalities between tasks, accelerate convergence to high-quality solutions, and enable effective knowledge transfer within evolutionary multi-tasking frameworks.

Moreover, future research could explore incorporating task-specific constraints or addressing dynamic optimisation problems where task environments evolve over time. By extending the capabilities of the surrogate model to handle these complexities, it could play a pivotal role in improving the efficiency and scalability of multi-tasking optimisation frameworks. Such advancements would not only enhance the practicality of the proposed model in real-world applications but also contribute to the broader development of multi-objective optimisation strategies.

Acknowledgments

This project is supported by Ningbo Digital Port Technologies Key Lab, Ningbo Science and Technology Bureau (Project ID 2023Z237 and 2022Z217), the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG3-RP-2022-031) and the Jiangsu Distinguished Professor Programme.

References

- [1] Reinhard Boysen and Anna Sergevna Bosman. 2024. Multi-objective Evolutionary Neural Architecture Search for Recurrent Neural Networks. *Neural Processing Letters* 56 (2024), 200. <https://doi.org/10.1007/s11063-024-11659-0>
- [2] Sinee Chaiyarin, Napassorn Rojbundit, Panichanok Piyabenjarad, Pimpatra Limpitigranon, Siraprapa Wisithipakdeekul, Pawaree Nonthasaen, and Paniti Achararit. 2024. Neural architecture search for medicine: A survey. *Informatics in Medicine Unlocked* 50 (2024), 101565. <https://doi.org/10.1016/j.imu.2024.101565>

- [3] Han Xiang Choong, Yew-Soon Ong, Abhishek Gupta, Caishun Chen, and Ray Lim. 2023. Jack and Masters of all Trades: One-Pass Learning Sets of Model Sets From Large Pre-Trained Models. *IEEE Computational Intelligence Magazine* 18, 3 (2023), 29–40. <https://doi.org/10.1109/MCI.2023.3277769>
- [4] Sumit Chopra, Raia Hadsell, and Yann LeCun. 2005. Learning a similarity metric discriminatively, with application to face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 539–546.
- [5] Xiangxiang Chu, Xiaoxing Wang, Bo Zhang, Shun Lu, Xiaolin Wei, and Junchi Yan. 2021. DARTS:: Robustly stepping out of performance collapse without indicators. In *International Conference on Learning Representations*.
- [6] Xiangxiang Chu, Bo Zhang, and Ruijun Xu. 2021. FairNAS: Rethinking evaluation fairness of weight sharing neural architecture search. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 12219–12228. <https://doi.org/10.1109/ICCV48922.2021.01202>
- [7] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6, 2 (2002), 182–197. <https://doi.org/10.1109/4235.996017>
- [8] Junwei Dong, Boyu Hou, Liang Feng, Huajin Tang, Kay Chen Tan, and Yew-Soon Ong. 2023. A Cell-Based Fast Memetic Algorithm for Automated Convolutional Neural Architecture Design. *IEEE Transactions on Neural Networks and Learning Systems* 34, 11 (2023), 9040–9053. <https://doi.org/10.1109/TNNLS.2022.3155230>
- [9] Xuanyi Dong and Yi Yang. 2019. One-shot neural architecture search via self-evaluated template network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 3680–3689. <https://doi.org/10.1109/ICCV.2019.00378>
- [10] Xuanyi Dong and Yi Yang. 2019. Searching for a robust neural architecture in four GPU hours. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 1761–1770. <https://doi.org/10.1109/CVPR.2019.00186>
- [11] Xuanyi Dong and Yi Yang. 2020. NAS-Bench-201: Extending the Scope of Reproducible Neural Architecture Search. In *International Conference on Learning Representations (ICLR)*. <https://openreview.net/forum?id=HJxyZkBKDr>
- [12] Stefan Falkner, Aaron Klein, and Frank Hutter. 2018. BOHB: Robust and efficient hyperparameter optimization at scale. In *Proceedings of the 35th International Conference on Machine Learning*, Vol. 80. 1437–1446.
- [13] Liang Fan and Handing Wang. 2023. Surrogate-assisted evolutionary neural architecture search with network embedding. *Complex & Intelligent Systems* 9, 3 (2023), 3313–3331. <https://doi.org/10.1007/s40747-022-00929-w>
- [14] Bryson Greenwood and Tyler McDonnell. 2022. Surrogate-assisted neuroevolution. In *Proceedings of the Genetic and Evolutionary Computation Conference (Boston, Massachusetts) (GECCO '22)*. Association for Computing Machinery, New York, NY, USA, 1048–1056. <https://doi.org/10.1145/3512290.3528703>
- [15] Bicheng Guo, Tao Chen, Shibo He, Haoyu Liu, Lilin Xu, Peng Ye, and Jiming Chen. 2022. Generalized global ranking-aware neural architecture ranker for efficient image classifier search. In *Proceedings of the 30th ACM International Conference on Multimedia*. 3730–3741. <https://doi.org/10.1145/3503161.3548149>
- [16] Pengcheng Jiang, Yu Xue, Ferrante Neri, and Mohamed Wahib. 2024. Surrogate-Assisted Evolutionary Neural Architecture Search with Isomorphic Training and Prediction. In *Advanced Intelligent Computing Technology and Applications*, De-Shuang Huang, Xiankun Zhang, and Yijie Pan (Eds.). Springer Nature Singapore, Singapore, 191–203.
- [17] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*, Yoshua Bengio and Yann LeCun (Eds.).
- [18] Nikita Klyuchnikov, Ilya Trofimov, Ekaterina Artemova, Mikhail Salnikov, Maxim Fedorov, Alexander Filippov, and Evgeny Burnaev. 2022. NAS-Bench-NLP: Neural Architecture Search Benchmark for Natural Language Processing. *IEEE Access* 10 (2022), 45736–45747. <https://doi.org/10.1109/ACCESS.2022.3169897>
- [19] Alex Krizhevsky. 2009. Learning Multiple Layers of Features from Tiny Images. <https://api.semanticscholar.org/CorpusID:18268744>
- [20] George Kyriakides and Konstantinos Margaritis. 2022. Evolving graph convolutional networks for neural architecture search. *Neural Computing and Applications* 34 (2022), 899–909. <https://doi.org/10.1007/s00521-021-05979-8>
- [21] Liam Li and Ameet Talwalkar. 2020. Random search and reproducibility for neural architecture search. In *Proceedings of the 35th Uncertainty in Artificial Intelligence Conference*. 367–377.
- [22] Yuhong Li, Cong Hao, Pan Li, Jinjun Xiong, and Deming Chen. 2021. Generic neural architecture search via regression. In *Advances in Neural Information Processing Systems*, Vol. 34. 20476–20490.
- [23] Hanxiao Liu, Karen Simonyan, and Yiming Yang. 2019. DARTS: Differentiable Architecture Search. *International Conference on Learning Representations (ICLR)* (2019).
- [24] Jiao Liu, Yew Soon Ong, and Melvin Wong. 2024. Finding Sets of Pareto Sets in Real-World Scenarios – A Multitask Multiobjective Perspective. In *2024 IEEE Congress on Evolutionary Computation (CEC)*. 1–8. <https://doi.org/10.1109/CEC60901.2024.10611967>
- [25] Shiqing Liu, Haoyu Zhang, and Yaochu Jin. 2022. A survey on computationally efficient neural architecture search. *Journal of Automation and Intelligence* 1, 1 (2022), 100002. <https://doi.org/10.1016/j.jai.2022.100002>
- [26] Zhichao Lu, Ran Cheng, Shihua Huang, Haoming Zhang, Changxiao Qiu, and Fan Yang. 2023. Surrogate-Assisted Multiobjective Neural Architecture Search for Real-Time Semantic Segmentation. *IEEE Transactions on Artificial Intelligence* 4, 6 (2023), 1602–1615. <https://doi.org/10.1109/TAI.2022.3213532>
- [27] Zhichao Lu, Ran Cheng, Yaochu Jin, Kay Chen Tan, and Kalyanmoy Deb. 2024. Neural Architecture Search as Multiobjective Optimization Benchmarks: Problem Formulation and Performance Assessment. *IEEE Transactions on Evolutionary Computation* 28, 2 (2024), 323–337. <https://doi.org/10.1109/TEVC.2022.3233364>
- [28] Zhichao Lu, Kalyanmoy Deb, Erik Goodman, Wolfgang Banzhaf, and Vishnu Naresh Boddeti. 2020. NSGANetV2: Evolutionary Multi-objective Surrogate-Assisted Neural Architecture Search. In *Computer Vision – ECCV 2020*, Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm (Eds.). Springer International Publishing, Cham, 35–51.
- [29] Kuangda Lyu, Hao Li, Maoguo Gong, Lining Xing, and A. K. Qin. 2024. Surrogate-Assisted Evolutionary Multiobjective Neural Architecture Search Based on Transfer Stacking and Knowledge Distillation. *IEEE Transactions on Evolutionary Computation* 28, 3 (2024), 608–622. <https://doi.org/10.1109/TEVC.2023.3319567>
- [30] Lianbo Ma, Nan Li, Guo Yu, Xiaoyu Geng, Shi Cheng, Xingwei Wang, Min Huang, and Yaochu Jin. 2024. Pareto-Wise Ranking Classifier for Multiobjective Evolutionary Neural Architecture Search. *IEEE Transactions on Evolutionary Computation* 28, 3 (2024), 570–581. <https://doi.org/10.1109/TEVC.2023.3314766>
- [31] Joseph Mellor, Jack Turner, Amos Storkey, and Elliot J Crowley. 2021. Neural architecture search without training. In *Proceedings of the 38th International Conference on Machine Learning*, Vol. 139. 7588–7598.
- [32] Gean T. Pereira, Iury B.A. Santos, Luis P.F. Garcia, Thierry Urruty, Muriel Visani, and André C.P.L.F. De Carvalho. 2023. Neural architecture search with interpretable meta-features and fast predictors. *Information Sciences* 649 (2023), 119642. <https://doi.org/10.1016/j.ins.2023.119642>
- [33] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. 2018. Efficient neural architecture search via parameters sharing. In *Proceedings of the 35th International Conference on Machine Learning*, Vol. 80. 4095–4104.
- [34] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le. 2017. Large-scale evolution of image classifiers. In *Proceedings of the International Conference on Machine Learning (ICML)*. 2902–2911.
- [35] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le. 2019. Regularized evolution for image classifier architecture search. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 587.
- [36] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. 2019. Regularized evolution for image classifier architecture search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 4780–4789. <https://doi.org/10.1609/aaai.v33i01.33014780>
- [37] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. 2020. Regularized Evolution for Image Classifier Architecture Search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 4780–4789.
- [38] Michael Ruchte, Arber Zela, Julien Siems, Josif Grabocka, and Frank Hutter. 2020. NASLib: A Modular and Flexible Neural Architecture Search Library. <https://github.com/automl/NASlib>.
- [39] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge. *arXiv:1409.0575 [cs.CV]* <https://arxiv.org/abs/1409.0575>
- [40] Jaeho Seong, Chaehyun Lee, and Dong Seog Han. 2022. Neural Architecture Search for Real-Time Driver Behavior Recognition. In *2022 International Conference on Artificial Intelligence in Information and Communication (ICAIIc)*. 104–108. <https://doi.org/10.1109/ICAIIc54071.2022.9722706>
- [41] Seyed Mahdi Sharatzadeh, Mahmood Fathy, Reza Berangi, and Mohammad Shahverdy. 2023. A Survey on Multi-Objective Neural Architecture Search. *ArXiv abs/2307.09099* (2023). <https://api.semanticscholar.org/CorpusID:259951144>
- [42] Yanan Sun, Xian Sun, Yuhuan Fang, Gary G. Yen, and Yuqiao Liu. 2021. A Novel Training Protocol for Performance Predictors of Evolutionary Neural Architecture Search Algorithms. *IEEE Transactions on Evolutionary Computation* 25, 3 (2021), 524–536. <https://doi.org/10.1109/TEVC.2021.3055076>
- [43] Chen Wei, Yiping Tang, Chuang Niu, Chuang Niu, Haihong Hu, Yue Wang, and Jimin Liang. 2021. Self-supervised representation learning for evolutionary neural architecture search. *IEEE Computational Intelligence Magazine* 16, 3 (2021), 33–49. <https://doi.org/10.1109/MCI.2021.3084415>
- [44] Wei Wen, Hanxiao Liu, Yiran Chen, Hai Li, Gabriel Bender, and Pieter-Jan Kindermans. 2020. Neural Predictor for Neural Architecture Search. In *Computer Vision – ECCV 2020*, Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm (Eds.). Springer International Publishing, Cham, 660–676.
- [45] Colin White, Willie Neiswanger, and Yash Savani. 2021. BANANAS: Bayesian optimization with neural architectures for neural architecture search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 10293–10301. <https://doi.org/10.1609/aaai.v35i12.17233>
- [46] Xiangning Xie, Yuqiao Liu, Yanan Sun, Gary G. Yen, Bing Xue, and Mengjie Zhang. 2022. BenchENAS: A Benchmarking Platform for Evolutionary Neural Architecture Search. *IEEE Transactions on Evolutionary Computation* 26, 6 (2022), 1473–1485. <https://doi.org/10.1109/TEVC.2022.3147526>

- [47] Yixing Xu, Yunhe Wang, Kai Han, Yehui Tang, Shangling Jui, Chunjing Xu, and Chang Xu. 2021. ReNAS: Relativistic Evaluation of Neural Architecture Search. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 4409–4418. <https://doi.org/10.1109/CVPR46437.2021.00439>
- [48] Yixing Xu, Yunhe Wang, Kai Han, Yehui Tang, Shangling Jui, Chunjing Xu, and Chang Xu. 2021. ReNAS: Relativistic evaluation of neural architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 4409–4418. <https://doi.org/10.1109/CVPR46437.2021.00439>
- [49] Yu Xue, Xiaolong Han, Ferrante Neri, Jiafeng Qin, and Danilo Pelusi. 2025. A Gradient-Guided Evolutionary Neural Architecture Search. *IEEE Transactions on Neural Networks and Learning Systems* 36, 3 (2025), 4345–4357. <https://doi.org/10.1109/TNNLS.2024.3371432>
- [50] Yu Xue, Weinan Tong, Ferrante Neri, Peng Chen, Tao Luo, Liangli Zhen, and Xiao Wang. 2024. Evolutionary Architecture Search for Generative Adversarial Networks Based On Weight Sharing. *IEEE Transactions on Evolutionary Computation* (2024), 1–1. <https://doi.org/10.1109/TEVC.2023.3338371> to appear.
- [51] Shangshang Yang, Ye Tian, Xiaoshu Xiang, Shichen Peng, and Xingyi Zhang. 2022. Accelerating Evolutionary Neural Architecture Search via Multifidelity Evaluation. *IEEE Transactions on Cognitive and Developmental Systems* 14, 4 (2022), 1778–1792. <https://doi.org/10.1109/TCDS.2022.3179482>
- [52] Xingyi Zhang, Ye Tian, Ran Cheng, and Yaochu Jin. 2015. An Efficient Approach to Nondominated Sorting for Evolutionary Multiobjective Optimization. *IEEE Transactions on Evolutionary Computation* 19, 2 (2015), 201–213. <https://doi.org/10.1109/TEVC.2014.2308305>
- [53] Barret Zoph and Quoc V Le. 2017. Neural Architecture Search with Reinforcement Learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*.
- [54] Barret Zoph and Quoc V Le. 2017. Neural architecture search with reinforcement learning. In *International Conference on Learning Representations*.
- [55] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. 2018. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1–14.