NSGA-Net: Neural Architecture Search using Multi-Objective Genetic Algorithm (Extended Abstract)*

Zhichao Lu, Ian Whalen, Yashesh Dhebar, Kalyanmoy Deb, Erik Goodman, Wolfgang Banzhaf and Vishnu Naresh Boddeti
Michigan State University
{luzhicha,whalenia,dhebarya,kdeb,goodman,banzhafw,vishnu}@msu.edu

Abstract
Convolutional neural networks (CNNs) are the backbones of deep learning paradigms for numerous vision tasks. Early advancements in CNN architectures are primarily driven by human expertise and elaborate design. Recently, neural architecture search (NAS) was proposed with the aim of automating the network design process and generating task-dependent architectures. This paper introduces NSGA-Net – an evolutionary search algorithm that explores a space of potential neural network architectures in three steps, namely, a population initialization step that is based on prior-knowledge from hand-crafted architectures, an exploration step comprising crossover and mutation of architectures, and finally an exploitation step that utilizes the hidden useful knowledge stored in the entire history of evaluated neural architectures in the form of a Bayesian Network. The integration of these components allows an efficient design of architectures that are competitive and in many cases outperform both manually and automatically designed architectures on CIFAR-10 classification task. The flexibility provided from simultaneously obtaining multiple architecture choices for different compute requirements further differentiates our approach from other methods in the literature.

1 Introduction
Deep convolutional neural networks have been overwhelmingly successful in a variety of computer-vision-related tasks like object classification, detection, and segmentation. One of the main driving forces behind this success is the introduction of many CNN architectures, including GoogLeNet [Szegedy et al., 2015], ResNet [He et al., 2016], DenseNet [Huang et al., 2017], etc., in the context of object classification. Concurrently, architecture designs such as ShuffleNet [Zhang et al., 2018], MobileNet [Sandler et al., 2018], LBCNN [Juefei-Xu et al., 2017], etc., have been developed with the goal of enabling real-world deployment of high-performance models on resource-constrained devices. These developments are the fruits of years of painstaking efforts and human ingenuity.

Neural architecture search (NAS), on the other hand, presents a promising path to alleviate this painful process by posing the design of CNN architectures as an optimization problem. By altering the architectural components in an algorithmic fashion, novel CNNs can be discovered that exhibit improved performance metrics on representative datasets. The huge surge in research and applications of NAS indicate the tremendous academic and industrial interest NAS has attracted, as teams seek to stake out some of this territory. It is now well recognized that designing bespoke neural network architectures for various tasks is one of the most challenging and practically beneficial component of the entire Deep Neural Network (DNN) development process, and is a fundamental step towards automated machine learning.

In this paper, we present NSGA-Net, a multi-objective genetic algorithm for NAS to address the aforementioned limitations of current approaches. A pictorial overview of NSGA-Net is provided in Figure 1. The salient features of NSGA-Net are, (1) multi-objective optimization: Real-world deployment of NAS models demands small-sized networks, in addition the models being accurate. For instance, we seek to maximize performance on compute devices that are often constrained by hardware resources in terms of power consumption, available memory, available FLOPs, and latency constraints, to name a few. NSGA-Net is explicitly designed to optimize such competing objectives. (2) Flexible architecture search space: The search space for most existing methods is restricted to a block that is repeated as many times as desired. In contrast, NSGA-Net searches over the entire structure of the network. This scheme overcomes the limitations inherent in repeating the same computation block throughout an entire network, namely, that a single block may not be optimal for every application and it is desirable to allow NAS to discover architectures with different blocks in different parts of the network. (3) Non-dominated sorting: The core component of NSGA-Net is the Non-Dominated Sorting and Crowding distance based multi-objective selection module from NSGA-II [Deb et al., 2002], a multi-objective optimization algorithm that has been successfully employed for solving a variety of multi-objective problems [Tapia and Coello, 2007; Pedersen and Yang, 2006]. Here, we leverage its ability to

*This is an extended abstract of the original paper [Lu et al., 2019], which won the best-paper award at the GECCO-2019 under Evolutionary Machine Learning track.
maintain a diverse trade-off frontier between multiple conflicting objectives, thereby resulting in a more effective and efficient exploration of the search space. (4) Efficient recombination: In contrast to state-of-the-art evolution-based NAS methods [Real et al., 2017; Real et al., 2019] in which only mutation is used, we employ crossover (in addition to mutation) to combine networks with desirable qualities across multiple objectives from the diverse frontier of solutions, and finally (5) Bayesian learning: We construct and employ a Bayesian Network inspired by the Bayesian Optimization Algorithm (BOA) [Pelikan et al., 1999] to fully utilize the promising solutions present in our search history archive and the inherent correlations between the layers of the network architecture. We demonstrate the efficacy of NSGA-Net on CIFAR10 [Krizhevsky et al., 2009] image classification task by minimizing two objectives: classification error and computational complexity. Here, computational complexity is defined by the number of floating-point operations (FLOPs) that a network carries out during a forward pass. Experimentally, we observe that NSGA-Net can find a set of network architectures containing solutions that are significantly better than hand-crafted methods in both objectives, while being competitive with single objective state-of-the-art NAS approaches. Furthermore, by fully utilizing a population of networks through recombination and utilization of the search history, NSGA-Net explores the search space efficiently and requires less computational time for search than other competing methods.

2 Proposed Approach

Compute devices are often constrained by hardware resources in terms of their power consumption, available memory, available FLOPs, and latency constraints. Hence, real-world design of DNNs are required to balance these multiple objectives (e.g., predictive performance and computational complexity). Often, when multiple design criteria are considered simultaneously, there may not exist a single solution that performs optimally under such circumstances, a set of solutions that provide representative trade-off information between the objectives is more desirable. This enables a practitioner to analyze the importance of each criterion, depending on the application, and to choose an appropriate solution on the trade-off frontier for implementation. We propose NSGA-Net, a genetic algorithm based architecture search method to automatically generate a set of DNN architectures that approximate the Pareto-front between performance and complexity on an image classification task. The rest of this section describes the encoding scheme, and main components of NSGA-Net in detail.

2.1 Encoding

Genetic algorithms, like any other biologically inspired search methods, often do not directly operate on phenotypes. From the biological perspective, we may view the DNN architecture as a phenotype, and the representation it is mapped from as its genotype. As in the natural world, genetic operations like crossover and mutation are only carried out in the genotype space; such is the case in NSGA-Net as well. We refer to the interface between the genotype and the phenotype as encoding in this paper. Most existing CNN architectures can be viewed as a composition of computational blocks that define the layer-wise computation (e.g., ResNet blocks [He et al., 2016], DenseNet block [Huang et al., 2017], and Inception block [Szegedy et al., 2015], etc.) and a scheme that specifies the spatial resolution changes. For example, down-sampling is often used after computational blocks to reduce the spatial resolution of information going into the next computational blocks in image classification DNNs. In NSGA-Net, each computational block, referred to as a phase, is encoded using the method presented by [Xie and Yuille, 2017], with the small change of adding a bit to represent a skip connection that forwards the input information directly to the output bypassing the entire block. And we name it as the Operation Encoding $x_o$ in this study.

Operation Encoding $x_o$. Unlike most of the hand-crafted and NAS generated architectures, we do not repeat the same phase (computational block) to construct a network. Instead, the operations of a network are encoded by $x_o = (x_o^{(1)}, x_o^{(2)}, \ldots, x_o^{(n_p)})$ where $n_p$ is the number of phases. Each $x_o^{(i)}$ encodes a directed acyclic graph consisting of $n_o$ number of nodes that describes the operation within a phase using a binary string. Here, a node is a basic computational unit, which can be a single operation like convolution, pooling, batch-normalization [Ioffe and Szegedy, 2015] or a sequence of operations. This encoding scheme offers a compact representation of the network architectures in genotype space, yet is flexible enough that many of the computational blocks in hand-crafted networks can be encoded, e.g. VGG [Simonyan and Zisserman, 2015], ResNet [He et al., 2016] and DenseNet [Huang et al., 2017]. Figure 2 and Figure 3 shows examples of the operation encoding.

Search Space. With a pre-determined scheme of spatial resolution reduction (similarly in [Zoph et al., 2018; Real et al.,
2.2 Search Procedure

NSGA-Net is an iterative process in which initial solutions are made gradually better as a group, called a population. In every iteration, the same number of offspring (new network architectures) are generated from parents selected from the population. Each population member (including both parents and offspring) compete for both survival and reproduction (becoming a parent) in the next iteration. The initial population may be generated randomly or guided by prior-knowledge (e.g. seeding the hand-crafted network architectures into the initial population). Following initialization, the overall NSGA-Net search proceeds in two sequential stages, an exploration and exploitation.

**Exploration.** The goal of this stage is to discover diverse ways of connecting nodes to form a phase (computational block). Genetic operations, crossover and mutation, offer an effective mean to realize this goal.

**Crossover.** The implicit parallelism of population-based search approaches can be unlocked when the population members can effectively share (through crossover) building-blocks [Holland, 1975]. In the context of NAS, a phase or the sub-structure of a phase can be viewed as a building-block. We design a homogeneous crossover operator, which takes two selected population members as parents, to create offspring (new network architectures) by inheriting and recombining the building-blocks from parents. The main idea of this crossover operator is to 1) preserve the common building-blocks shared between both parents by inheriting the common bits from both parents’ binary bit-strings; 2) maintain, relatively, the same complexity between the parents and their offspring by restricting the number of “1” bits in the offspring’s bit-string to lie between the number of “1” bits in both parents. The proposed crossover allows selected architectures (parents) to effectively exchange phases or sub-structures within a phase. An example of the crossover operator is provided in Figure 3.

**Mutation.** To enhance the diversity (having different network architectures) of the population and the ability to escape...
from local optima, we use a bit-flipping mutation operator, which is commonly used in binary-coded genetic algorithms. Due to the nature of our encoding, a one bit flip in the genotype space could potentially create a completely different architecture in the phenotype space. Hence, we restrict the number of bits that can be flipped to be at most one for each mutation operation. As a result, only one of the phase architectures can be mutated at one time.

**Exploitation.** The exploitation stage follows the exploration stage in NSGA-Net. The goal of this stage is to exploit and reinforce the patterns commonly shared among the past successful architectures explored in the previous stage. The exploitation step in NSGA-Net is heavily inspired by the Bayesian Optimization Algorithm (BOA) [Pelikan et al., 1999] which is explicitly designed for problems with inherent correlations between the optimization variables. In the context of our NAS encoding, this translates to correlations in the blocks and paths across the different phases. Exploitation uses past information across all networks evaluated to guide the final part of the search. More specifically, say we have a network with three phases, namely $x_0^{(1)}$, $x_0^{(2)}$, and $x_0^{(3)}$. We would like to know the relationship of the three phases. For this purpose, we construct a Bayesian Network (BN) relating these variables, modeling the probability of networks beginning with a particular phase $x_0^{(1)}$, the probability that $x_0^{(2)}$ follows $x_0^{(1)}$, and the probability that $x_0^{(3)}$ follows $x_0^{(2)}$. In other words, we estimate the distributions $p(x_0^{(1)})$, $p(x_0^{(2)}|x_0^{(1)})$, and $p(x_0^{(3)}|x_0^{(2)})$ by using the population history, and update these estimates during the exploitation process. New offspring solutions are created by sampling from this BN.

### 3 Experiments

In this section, we briefly explain the experimental setup and implementation details of NSGA-Net, followed by the empirical results to demonstrate the efficacy of NSGA-Net to automate the NAS process on image classification task in Table 1.

#### Dataset

We consider the CIFAR-10 [Krizhevsky et al., 2009] dataset for our classification task. We split the original training set (80%-20%) to create our training and validation sets for architecture search. The original CIFAR-10 testing set is only utilized at the conclusion of the search to obtain the test accuracy for the models on the final trade-off front.

#### Hyper-parameters

We set the number of phases $n_p$ to three and the number of nodes in each phase $n_o$ to six. The initial population is generated by uniform random sampling. The probabilities of crossover and mutation operations are set at 0.9 and 0.02 respectively. The population size is 40 and the number of generations is 20 for the exploration stage. And another ten generations for exploitation. Hence, a total of 1,200 network architectures are searched by NSGA-Net. During architecture search, we limit the number of filters (channels) in any node to 16 for each one of the generated network architecture. We then train them on our training set using stochastic gradient descent (SGD) with initial learning rate is 0.025 anneals down to zero in 25 epochs.

### 4 Conclusions

This paper presented NSGA-Net, a multi-objective evolutionary approach for neural architecture search. NSGA-Net affords a number of practical benefits: (1) the design of neural network architectures that can effectively optimize and trade-off multiple competing objectives, (2) advantages afforded by population-based methods being more effective than optimizing weighted linear combination of objectives, and (3) more efficient exploration and exploitation of the search space through a customized crossover scheme and leveraging the entire search history through BOA. Experimentally, by optimizing both prediction performance and computational complexity NSGA-Net finds networks that are significantly better than hand-crafted networks on both objectives and is compares favorably to other state-of-the-art single objective NAS methods for classification on CIFAR-10.
References


[Pham et al., 2018] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. Efficient neural architecture search via parameters sharing. In ICML, 2018.


