Neural Network Evolution Using Expedited Genetic Algorithm for Medical Image Denoising

Peng Liu, Yangjunyi Li, Mohammad D. El Basha, and Ruogu Fang(✉)

J. Crayton Pruitt Family Department of Biomedical Engineering,
University of Florida, Gainesville, FL, USA
Ruogu.Fang@bme.ufl.edu

Abstract. Convolutional neural networks offer state-of-the-art performance for medical image denoising. However, their architectures are manually designed for different noise types. The realistic noise in medical images is usually mixed and complicated, and sometimes unknown, leading to challenges in creating effective denoising neural networks. In this paper, we present a Genetic Algorithm (GA)-based network evolution approach to search for the fittest genes to optimize network structures. We expedite the evolutionary process through an experience-based greedy exploration strategy and transfer learning. The experimental results on computed tomography perfusion (CTP) images denoising demonstrate the capability of the method to select the fittest genes for building high-performance networks, named EvoNets, and our results compare favorably with state-of-the-art methods.

Keywords: Medical image denoising · Genetic Algorithm
Convolutional neural networks · Evolution · Low-dose imaging

1 Introduction

Medical imaging techniques, such as Computed Tomography (CT), Magnetic Resonance Imaging (MRI), and X-rays are popular diagnostic tools. Nevertheless, these techniques are susceptible to noise. For example, CT perfusion images are often associated with complicated mixed noise due to the photon starvation artifacts. In recent decades, different methods have been widely investigated to solve the problem, ranging from spatial filtering techniques, such as Wiener filters [7], to patch similarity methods, such as BM3D [1]. However, complicated mixed noise in medical images still leads to the unsatisfactory performance of these methods and remains a valuable research direction.

Convolutional Neural Networks (CNN) have shown superior performance over traditional models on denoising tasks. A typical CNN is composed of several stacked layers, including layer connections and hyperparameters (e.g., number of layers, neurons in each layer, type of activation function). RED-Net [5] consists of a chain of 30 convolutional layers and symmetric deconvolutional layers.

A. F. Frangi et al. (Eds.): MICCAI 2018, LNCS 11070, pp. 12–20, 2018.
https://doi.org/10.1007/978-3-030-00928-1_2

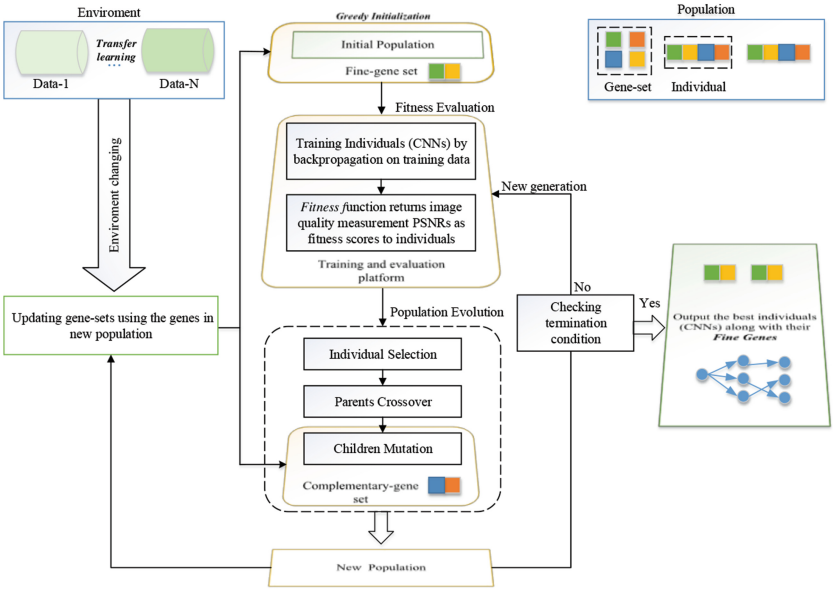

Fig. 1. Overview of the proposed method composing of fitness evaluation and population evolution. The CNN architecture is trained on medical images using a fitness score. The individual networks labeled with the fitness scores are sent to individual selection. The survived individuals are presented as parents for crossover and mutation.

Another state-of-the-art method, DnCNN [10], adopts concise stacked-layer connections but achieves impressive performance via appropriate hyperparameters (e.g., ReLU [6]) selection. Consequently, hyperparameters play a dominant role in optimizing image denoising tasks.

Although these modern networks present promising image restoration performance, they are all manually designed based on empirical knowledge. It is expensive and slow to manually search for the optimal network structures with exponential combinations of hyperparameters and layers connections. To address this issue, it is critical to automatically construct promising CNN-based denoisers with concise layer connections and optimal hyperparameter combinations. Moreover, an efficient algorithm is important to explore the optimal CNN structures within reasonable computational time.

In this work, we construct a CNN-based medical image denoiser, named EvoNet, automatically. To more effectively navigate large search spaces, we formulate an optimized genetic algorithm (GA). Basically, GA initializes candidate solutions (e.g., networks) as an initial generation, and then applies genetic operations to evolve the solutions in each generation. As shown in Fig. 1, for the population evolution process, we define three standard genetic operations: selection, crossover, and mutation. A fitness function is formulated to help us select best individuals (e.g., CNN) in each generation. Each of these solutions is

evaluated by fitness scores through a denoising evaluation criteria. The contributions of the paper are as follows:

- It is the first time to propose a GA-based method to construct CNN structures for medical image denoising automatically. This evolution approach provides the flexibility to optimize both CNN parameters and network structures.
- We optimize the standard genetic algorithm to speed up the evolutionary progress. Specifically, we use an experience based greedy strategy on the initialization stage to enrich high-performance individuals in the first generation. In addition, we select an appropriate mutation rate to make a trade-off between the diversity of the population (CNNs) and convergence of optimum generation.
- We dynamically update hyperparameter sets to make the architectures of the population (CNNs) transferable between datasets of different sizes. Particularly, we split all possible hyperparameters into fine-genes and complementary-genes for initialization and mutation respectively.

2 Methodology

Background. Genetic Algorithms (GAs) [2] are inspired by the natural biological evolution. Typically, a GA is composed of a "population" P of N "individuals", and has operations including initialization, individual selection, parents crossover, and children mutation (see Fig. 1). A sequence of operations is referred as an evolutionary "generation". The competition among individuals is simulated by a *fitness* function that selects the fittest individuals over the weaker ones.

GA has been widely utilized as a heuristic search and optimization technique and also has been applied in machine learning approaches, and function optimization. Recently, Xie et al. [8] applied GA to explore CNN architectures automatically for image classification. These methods focus on exploring the structural module blocks and connections among layers. However, the study of an efficient way for building a concise CNN-based denoiser on medical image automatically is still lacking.

A concise but also promising CNN-based denoiser relies on a specific learning strategy (e.g., Residual learning) and one choice of hyperparameter combinations (e.g., DnCNN). Therefore, in this work, we aim at building a simple but effective CNN structure via focusing on exploring the effective combinations of CNN hyperparameters instead of the structural blocks and layer connections. One significant challenge of using GA is how to accelerate the evolutionary process dynamically in a huge search space. To address this issue, we present an *Optimized Genetic Algorithm* (Algorithm 1) with an experience based greedy exploration strategy in the next section.

Gene Splitting. A "gene" is the basic functional unit in a biological body. In an artificial neural network, genes represent hyperparameters, such as the number of layers, the number of neurons, the activation function, and the type

Algorithm 1. The Proposed Genetic Algorithm for Exploring CNNs

Input: one all-possible-gene set $\theta = \theta_c \cup \theta_f$, initial fine-gene set θ_f, initial complementary-gene set θ_c, initial population size N, initial number of generation G, percentage of selected individuals after each generation σ, number of children of crossed over out O, mutation rate ϵ, termination condition E, small and large training datasets $D = \{D_s, D_l\}$

for d = 1, 2, ..., length(D) **do**
  for g = 1, 2, ..., $G-1$ **do**
    for i = 1, 2, ..., N **do**
      if $g = 1$ **then**
        Initialize a set of randomized individuals $\{P_i^g\}_{i=1}^N$ based on θ_f
      end if
      Return *trained* individuals $\{P_i^{g,t}\}_{i=1}^N$ by Keras and Tensorflow
      Return *fitness scores* $F_i^g = F(P_i^{g,t})$ to individuals
    end for
    $Sort\left\{P_i^{g,trained}\right\}_{i=1}^N$ by F_i^g with descending order
    $\Re = \left\{P_i^{g,trained,sorted}\right\}_{i=1}^{N*\sigma}$ *Select the top* $N*\sigma$ *best individuals*
    $P_i^{g,new} = P_i^{g,new} += \varnothing$
    while $length(P_i^{g,new}) < N - length(\Re)$ **do**
      $\Upsilon_{mom}, \Upsilon_{dad} = uniformRandom(\Re)$ Select parents
      $\{\Psi_o\}_{o=1}^O = Genome(\Upsilon_{mom}, \Upsilon_{dad})$ Have children with crossover genes
      if $\epsilon > Random(0,1)$ **then** Mutation with a rate μ
        $\Psi_{selected} = selectRandom(\{\Psi_o\}_{o=1}^O)$ Randomly select one child
        $\theta_{selected} = selectRandom(\theta_{\Psi_{selected}})$ Randomly select one gene
        $\theta_{c,selected} = selectRandom(\theta_c, Genotype(\theta_{selected}))$
        $\{\Psi_o^m\}_{o=1}^O = replace(\Psi_{selected}, \theta_{selected}, \theta_{c,selected})$
        $P_i^{g,new} = P_i^{g,new} + \{\Psi_o^m\}_{o=1}^O$
      else
        $P_i^{g,new} = P_i^{g,new} + \{\Psi_o\}_{o=1}^O$
      end if
      $P_i^{g,new} = removeDuplicate(P_i^{g,new})$
    end while
    $P_i^{g,new} = P_i^{g,new} + \Re$
    if $E = True$ **then** May say "the highest fitness score is not changing"
      Terminate generation and go to output
    end if
  end for
  $\theta_f^u = Update(\theta_f, \Re)$ Replace fine-gene set with the genes in $\Re$
  $\theta_c^u = \theta - \theta_f$ Update complementary-gene set
end for

Output: Select the best individuals (CNNs) from $P_i^{g,new}$

of optimizers. To speed up the evolution process, let θ be the set of all possible genes, and it is split into a fine-gene set θ_f and a complementary-gene set θ_c. Fine-genes are the hyperparameters selected from those state-of-the-art CNN structures in the literature (e.g., DnCNN) or previous GA generations. The rest

genes in θ are the complementary genes. The first population is initialized based on θ_f. The mutation process is solely built upon θ_c. An individual (CNN) is composed of different genes, and N individuals form a population-P.

Our method emphasizes the fittest gene more than the survived individuals (network structures). This strategy ensures the promising genes are passed down to offspring, and the fittest individuals are more likely to be explored effectively in early generations. Therefore, our approach can accelerate the evolution process via optimizing gene search space dynamically. The overview and algorithm details of the proposed method are shown in Fig. 1 and Algorithm 1 respectively.

Experience Based Greedy Exploration. We optimize GA with an experience based greedy exploration strategy, which determines how to update gene sets and terminate evolution process. Experience represents CNN hyperparameters from the last generation, our approach stores and transfers such experience to next generation. In another word, we initialize the fine-gene sets with top-performance CNNs evolved in the previous generations.

Transfer Learning. Another novel contribution of our approach is using a *transfer learning* strategy [9] that allows the explored CNN architectures to be transferable among training data of different sizes. For instance, we may use a small dataset to quickly optimize the gene-set space first, and then explore CNNs on a larger dataset by initializing a new population using the fine genes identified from the small dataset. It further expedites the network evolution process.

Fitness Evaluation. The *fitness* function $F(P_i)$ returns the restored image quality measure as a fitness score to each individual P_i. Fitness score performs the following functions: (1) evaluating individual fitness; (2) updating gene-sets; (3) serving as a stopping rule. Hence, the fitness function is critical for designing an effective GA-based method. Algorithm 1 presents the details of the proposed GA for exploring the promising CNNs to handle with medical image denoising.

3 Experiments

Training and Testing Data. Our dataset is a collection of 10,775 cerebral perfusion CT images, all of which are 512 $\times$ 512 gray-scale images. Training data D consist of randomly selected 250 images from the perfusion CT dataset, all of them are cropped uniformly to the size of 331 $\times$ 363. This pre-processing step removes skull and background from raw CT images and improves feature learning efficiency during training. Testing data are randomly selected 250 images with no overlap with the training data, and they remain as 512 $\times$ 512 grayscale images. Another 100 images with no overlap with the training/testing data are selected as the validation set. We use Peak Signal-to-Noise Ratio (PSNR) as the fitness function in approach.

Transfer Learning. GA requires high computational resources due to the large search space, which leads difficulties to evaluate performance on large datasets directly. Our strategy is to explore promising CNN hyperparameter combinations by training on a small subset D_s. In particular, 35 images from the training data are randomly selected and segmented with patch size 50×50 at a stride of 20. Therein, 8,576 image patches are generated for the initial evolution. We then transfer hyperparameters observed from results on D_s to a large training set D_l. With the same patch size and stride length, 100 images of D_l are segmented into 17,280 patches for further evolution.

Low-Dose Noise Simulation. Repeated scans at different radiation dose on the same patient are not ethical due to increased unnecessary radiation exposure. Therefore, in this paper, low-dose perfusion CT images are stimulated and added to the regular dose perfusion CT images. Specifically, spatially correlated, normally distributed noise is added to both training data and testing data. The added noise has a standard deviation of $\sigma = 17, 22, 32$, which corresponds to the tube current-time product of 30, 20, 10 mAs. The regular dose level is 190 mAs.

Experimental Setup. All possible genes θ are selected from CNN hyperparameters with promising performance reported in the literature [5,10]. In this paper, we consider a constrained case with θ consisting of four sub-genotypes: number of layers = (1, 2, 3, 4, 5, 6, 7), number of neurons in each layer = (16, 32, 64, 96, 128, 256), activation = ('ReLU', 'Tanh', 'SELU', 'ELU', 'Sigmoid'), and optimizers = ('rmsprop', 'sgd', 'adam','adamax', 'adadelta', 'adagrad'). During initialization, we set the initial fine-gene set θ_f from set θ as number of layers = (5, 6), number of neurons in each layer = (32, 48), activation = ('ReLU', 'ELU', 'Sigmoid'), and optimizers = ('sgd','adam'). We create an initial population size $N = 20$ individuals and perform genetic operations for 10 rounds (generation). For each generation, we set mutation possibility rate $\epsilon = 0.1$. Crossover happens between any two random parents networks. After each crossover and muta-

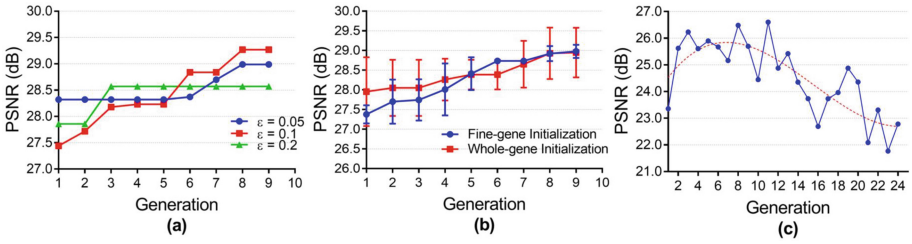

Fig. 2. (a) The performance of best individual with respect to mutation rate $\epsilon = 0.05, 0.1, 0.2$. (b) The average performance over top 5 individuals with respect to the initialization process with a fine-gene set θ_f and whole-gene set θ. (c) The average performance overall individuals with respect to the generation number. All training are processed on large dataset D_l

tion, we check the whole population and eliminate duplicate individuals (see Algorithm 1). Other hyperparameters (e.g., learning rate) follow Tensorflow default settings. Residual learning [4] is adopted to accelerate training process. All GA progresses are processed on Tensorflow platform with GEFORCE GTX TITAN GPUs.

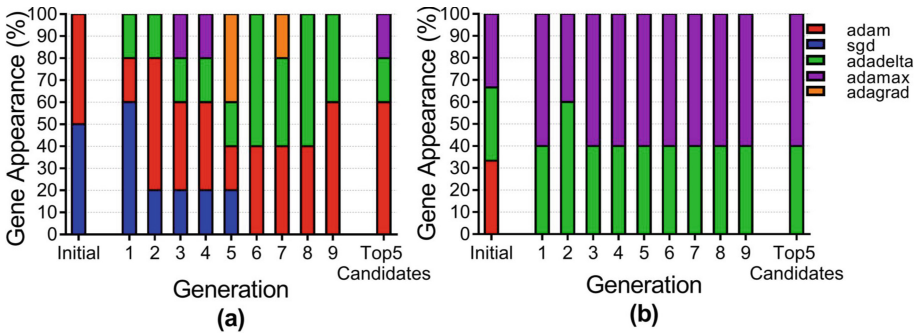

Fig. 3. The activation gene appearance changing during evolutionary progress on the small dataset D_s (a), and the large dataset D_l (b) with transferred initialization set. In each generation, 5 top performed individuals are selected to summarize changes. Top 5 Candidates bar refers to the final optimizer gene distribution after one evolutionary progress.

Parameters Selection. We evaluate the performance of different mutation rate as shown in Fig. 2(a). When the mutation rate is too high, it increases the searching speed in the search space but may not find optimal individuals in each generation. On the other hand, when the mutation rate is too low, it can lead individuals to converge rapidly to local optimum instead of the global optimum. From Fig. 2(a), $\epsilon = 0.1$ gives the optimal performance. We also evaluate different initialization strategies as shown in Fig. 2(b). Fine-gene initialization with selected genes can reach the same performance as the whole gene initialization strategy after 8 generations. While we set fine-genes as greedy initialization set, it helps early generations find high-performance individuals. However, after certain generations, more mutation genes are introduced due to the duplicate individual elimination, which increases population diversity but reduces the average performance. This strategy helps to stop early at an optimal generation and improves search efficiency. This is demonstrated in Fig. 2(c). We use 10 generations as shown in Fig. 2(c).

Gene Evolution. We track the evolution of genes over generations and illustrate the optimizer genes in Fig. 3. We show the top 5 individuals in each generation trained on a small training set and after transferring to a large training set. When training on a small set (Fig. 3(a)), the low-performance genes are eliminated over the generations, such as *sgd* and *adagrad*. At the same time,

the high-performance genes are introduced from mutation, such as *adadelta*. After being transferred to a large training set (Fig. 3(b)), the initialization set is transferred from (a), where good "genes" such as *adam, adadelta*, and *adamax* are preserved. Through the evolution, top performance genes such as *adamax* and *adadelta* dominate the optimizer genes. This tracking process demonstrates that our greedy initialization strategy helps to search for high-performance genes efficiently. More importantly, it shows that the learned CNN hyperparameters (genes) and structures are transferable from small datasets to large datasets.

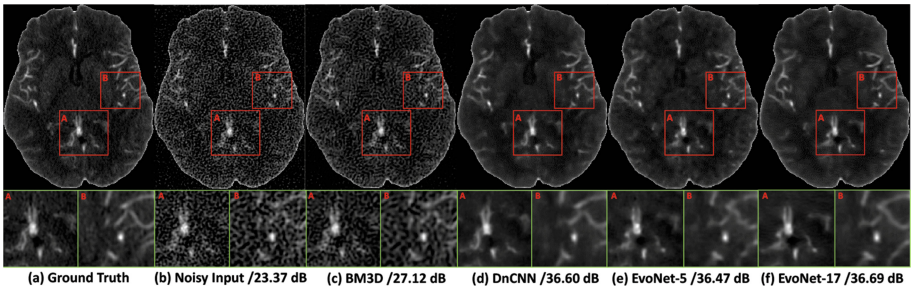

Fig. 4. Visual Results of perfusion CT dataset with noise $\sigma = 22$. A region of Interest (ROI) is selected (red region) and scaled up for better visual comparison.

Comparison with State-of-the-Art Methods. Both quantitative and qualitative comparisons are provided. We compared with state-of-the-art methods including BM3D and DnCNN. DnCNN has been reported to work on medical images [3]. We obtained the EvoNet-5 (5 layers, 64 neurons each layer, *adadelta*, *ReLu*) from D_s, and EvoNet-17 (17 layers, 64 neurons each layer, *adadelta*, *ReLu*) from D_l.

In Table 1, we present the summary of quantitative results. The deeper EvoNet-17 outperforms other state-of-the-art methods with PSNR on the testing dataset. The shallow EvoNet-5 achieves comparable performance to DnCNN; however, it is deep (20 layers) while the EvoNet-5 is a compact structure with

Table 1. Average PSNR, SSIM, and computation time of algorithms: BM3D, DnCNN, EvoNet-5, and EvoNet-17 at different noise levels $\sigma = 17, 22, 32$. Best performance is highlighted in bold.

σ	BM3D [1]		DnCNN [10]		EvoNet-5		EvoNet-17	
	PSNR(dB)	SSIM	PSNR(dB)	SSIM	PSNR(dB)	SSIM	PSNR(dB)	SSIM
17	29.07	0.4515	36.64	**0.9158**	36.30	0.9062	**36.65**	0.9074
22	26.98	0.3578	35.87	0.8863	35.66	0.8914	**35.92**	**0.8988**
32	23.95	0.2385	35.03	0.8671	34.35	0.8578	**35.04**	**0.8846**

stacked convolutional layers without regularization technique. Deeper (6, 7 layers) and larger (128, 256 neurons) networks are eliminated due to overfitting on small data. Figure 4 shows visual results. Our method perfectly restores physiological structures, circuit contour and texture of the cerebral cortex and gains high PSNR values. It is matching with quantitative results.

4 Conclusions

In this work, we propose an optimized GA-based strategy to explore CNN structure for medical image denoising. We introduce an experience-based greedy exploration strategy and transfer learning to accelerate GA evolution. We evaluate EvoNets on a perfusion CT dataset and demonstrate promising performance. In the current work, we only consider a constrained case. In future work, the proposed method can be extended to explore more flexible CNN structures for challenging tasks, such as tumor detection.

Acknowledgment. This work is partially supported by NSF IIS-1564892.

References

1. Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K.: BM3D image denoising with shape-adaptive principal component analysis. In: SPARS 2009-Signal Processing with Adaptive Sparse Structured Representations (2009)
2. Holland, J.H.: Genetic algorithms. Sci. Am. **267**(1), 66–73 (1992)
3. Jifara, W., Jiang, F., Rho, S., Cheng, M., Liu, S.: Medical image denoising using convolutional neural network: a residual learning approach. J. Supercomput. pp. 1–15 (2017)
4. Kiku, D., Monno, Y., Tanaka, M., Okutomi, M.: Residual interpolation for color image demosaicking. In: 2013 IEEE International Conference on Image Processing, pp. 2304–2308. IEEE (2013)
5. Mao, X.J., Shen, C., Yang, Y.B.: Image restoration using convolutional auto-encoders with symmetric skip connections. arXiv preprint arXiv:1606.08921 (2016)
6. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10), pp. 807–814 (2010)
7. Wintermark, M., Lev, M.: FDA investigates the safety of brain perfusion CT. Am. J. Neuroradiol. **31**(1), 2–3 (2010)
8. Xie, L., Yuille, A.: Genetic cnn. arXiv preprint arXiv:1703.01513 (2017)
9. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? In: Advances in Neural Information Processing Systems, pp. 3320–3328 (2014)
10. Zhang, K., Zuo, W., Chen, Y., Meng, D., Zhang, L.: Beyond a gaussian denoiser: residual learning of deep cnn for image denoising. IEEE Trans. Image Proc. **26**(7), 3142–3155 (2017)