Neural Architecture Search for Adversarial Medical Image Segmentation

Chapter · October 2019

DOI: 10.1007/978-3-030-32226-7_92

Min Xu
Carnegie Mellon University

Xiaodan Liang
Carnegie Mellon University

Neural Architecture Search for Adversarial Medical Image Segmentation

Nanqing Dong ✉[1,2], Min Xu[1,3], Xiaodan Liang[1,4], Yiliang Jiang[5], Wei Dai[6], and Eric Xing[1]

[1] Petuum, Inc., Pittsburgh, USA
[2] University of Oxford, Oxford, UK
`nanqing.dong@cs.ox.ac.uk`
[3] Carnegie Mellon University, Pittsburgh, USA
[4] Sun Yat-sen University, Guangzhou, China
[5] New York University, New York City, USA
[6] Apple Inc., Seattle, USA

Abstract. Adversarial training has led to breakthroughs in many medical image segmentation tasks. The network architecture design of the adversarial networks needs to leverage human expertise. Despite the fact that discriminator plays an important role in the training process, it is still unclear how to design an optimal discriminator. In this work, we propose a neural architecture search framework for adversarial medical image segmentation. We automate the process of neural architecture design for the discriminator with continuous relaxation and gradient-based optimization. We empirically analyze and evaluate the proposed framework in the task of chest organ segmentation and explore the potential of automated machine learning in medical applications. We further release a benchmark dataset for chest organ segmentation.

Keywords: Neural Architecture Search, Adversarial Networks, Medical Image Segmentation

1 Introduction

Inspired by the *generative adversarial networks* (GANs) [4], adversarial training for semantic segmentation was first proposed in [9], where an auxiliary discriminator is introduced to distinguish between the ground truth, and the segmentation output and the segmentation network is trained to fool the discriminator. A recent cognitive study implies that *convolutional neural networks* (CNNs) are more sensitive to the local texture of an object than the global shape [1]. A well-designed discriminator is expected to learn high-order statistics of the objects [3], which is a complement to a stand-alone segmentation network. Compared with general objects, organs, anatomies, and tissues usually share similar representation among different patients, which controls the variance of the high-order statistics. For medical images, the segmentation network is expected to converge faster and output realistic and robust prediction under adversarial training [2].

Adversarial training has achieved state-of-the-art performance in many medical image segmentation tasks [3, 5, 10], but it is still unclear how to design adversarial networks. While a bad segmentation network can hinder the performance, a bad discriminator can sabotage the whole machine learning system. Human expertise and intuition still play important roles in designing the discriminator. Fueled by the advances in both deep learning and computer hardware, there is a growing interest in the study of *neural architecture search* (NAS), which automates the manual process of architecture design. Even though the models discovered by NAS have outperformed human-invented models in image classification and language modeling tasks [15, 16, 11, 7], NAS for adversarial networks and semantic segmentation is still underdeveloped. In addition, the underlying idea of NAS is that each dataset has a unique best-performing architecture. In the medical domain, data scarcity has been a long-standing challenge. The pixel-level ground truth requires manual annotation by the clinical professionals, which is often expensive to acquire. It is natural to leverage limited data by deploying an efficient model. An algorithmic solution to automated architecture design in adversarial training is desired by both the machine learning community and the medical image computing community.

In this work, we propose a NAS framework for adversarial medical image segmentation. The proposed framework will automatically find the architecture of a discriminator based on a segmentation network backbone. The automatically designed discriminator will improve the performance of the segmentation network through adversarial training. There are previous studies on NAS using reinforcement learning [15, 16, 11] or evolutionary algorithm [14]. These methods require a huge amount of computational power when searching over the discrete domain, which is impractical in adversarial training. We develop our method based on *differentiable architecture search* [7], which relaxes the search space from discrete to continuous and can be optimized through gradient descent. Compared with a single end-to-end network in [7], we take the mutual effect between two adversarial networks into consideration during the alternative optimization process. To the best of our knowledge, this is the first study on NAS in adversarial training and the first application of NAS in the medical domain. Adversarial training is a complex non-convex optimization problem. Currently, we cannot mathematically prove that the proposed method can find the best architecture of the discriminator. Here, we treat this as a black-box optimization problem and approximate the optimal solution numerically. We empirically analyze and evaluate the proposed method in the task of chest organ segmentation on two datasets. We choose chest organ segmentation because it is a representative task in medical image segmentation that can be solved by adversarial training. Besides, chest organ segmentation has less computational cost than other complicated tasks, which allows for fast prototyping under limited computational power. The experiments show that our method can automate the process of architecture design in adversarial training and achieve comparable performance. The framework can be further extended to more complicated medical image segmentation tasks.

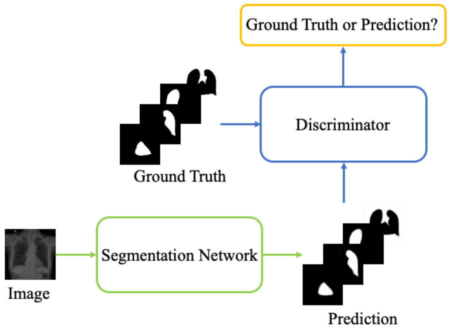

Fig. 1: Adversarial Semantic Segmentation for Chest Organs.

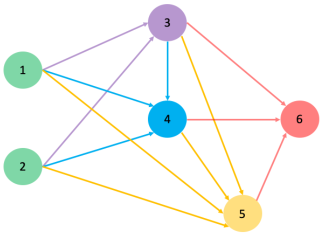

Fig. 2: An example of topological graph when $B = 6$. Green nodes are the input nodes and red node is the output node.

2 Method

In standard adversarial training for semantic segmentation, there is one segmentation network S and one discriminator D. S is a *fully convolutional network* (FCN) [8], which is parameterized as f_θ, and D is a CNN for binary classification, which is parameterized as g_ϕ. S is trained to output prediction realistic enough to confuse D, which in turn tries to discriminate the prediction from ground truth. A pipeline for chest organ segmentation is shown in Fig. 1. Let x denote the input image and y denote the corresponding ground truth. S and D are optimized alternatively. Given ϕ fixed, θ is updated by minimizing

$$\mathcal{L}(\theta) = \mathcal{L}_{seg}(f_\theta(x), y) - \lambda_{adv} \log g_\phi(f_\theta(x)) \tag{1}$$

, where $\mathcal{L}_{seg}$ is the softmax cross-entropy and λ_{adv} controls the weight of adversarial loss. Given θ fixed, ϕ is updated by minimizing

$$\mathcal{L}(\phi) = -\log g_\phi(y) - \log(1 - g_\phi(f_\theta(x)). \tag{2}$$

2.1 Problem Formulation

The focus of this work is to study how to automatically design the architecture of D. The intuition behind is that, given a dataset, for $\forall S \in \{\mathcal{S}\}$, $\exists D \in \{\mathcal{D}\}$, where (S, D) can have optimal performance under adversarial training. Here, to study the behavior of D alone, we fix the architecture of S. Assume the architecture of D can be parameterized as α by treating the architecture as a hyperparameter [7]. So D can be represented as $g_{\alpha,\phi}$. The goal is to jointly learn (α, θ, ϕ). To easily understand the relationships among (α, θ, ϕ), we decompose the problem into two stages. If we feed D with inputs independent of S, the problem becomes a standard NAS for CNN. For an isolated D, ϕ is associated with α. Then, by taking (α, ϕ) as a whole, S and D have mutual effects during the adversarial training, as stated in Eq. 1 and 2. Note, the relationship here does not imply any statistical dependency.

Here, we propose our method. We transform the problem into a bilevel optimization problem [7]. Given α fixed, the problem becomes a standard adversarial training, where we already know how to jointly optimize (θ, ϕ). Given (θ, ϕ) fixed, we can then optimize α, if the parameterization of α can be independent of ϕ. As a standard practice in NAS [15, 16, 11, 7], the training data is split into *train* and *val*. *train* is used to learn the weights and *val* is used to meta-learn the architecture. Let $\mathcal{J}$ denote certain joint optimization objective in adversarial training in Eq. 1 and 2. Assume $\mathcal{J}$ can be minimized to provide the best segmentation performance, the optimization goal for architecture search is to find α^* that minimizes $\mathcal{J}_{val}(\alpha^*, \theta^*, \phi^*)$, where the (θ^*, ϕ^*) are obtained by minimizing $\mathcal{J}_{train}(\alpha^*, \theta, \phi)$. Analogous to [7], we have

$$\min_{\alpha} \mathcal{J}_{val}(\alpha, \theta^*, \phi^*), \tag{3}$$

$$\text{where } \theta^*, \phi^* = \underset{\theta, \phi}{\operatorname{argmin}}\, \mathcal{J}_{train}(\alpha, \theta, \phi). \tag{4}$$

2.2 Neural Architecture Search Setting

It is impractical to search the model architecture starting from scratch, which can take more than 2000 GPU days for a simple classification task [15]. Here, we follow the basic setting of NAS introduced by [16, 11, 7]. We restrict the search space by searching the architecture of the computation *cell*. The cell is the basic unit, which can be stacked multiple times to form a CNN. A cell can be viewed as a directed acyclic graph which consists of B ordered nodes. The goal is to search the topology of the cell. Each node(vertex) v^i is a feature map and each directed edge (i, j) is certain operation $e^{(i,j)}$ applied to v^i. Each cell has two input nodes and one output node. The input nodes are the output nodes of two previous cells and the output node concatenates all intermediate nodes within the cell. The intermediate node takes the sum over all previous nodes.

$$v^j = \sum_{i<j} e^{(i,j)}(v^i) \tag{5}$$

An example topology is illustrated in Fig. 2.

As discussed in Sec. 2.1, the parameterization of α is required to be independent of ϕ. We adopt *continuous relaxation* proposed by [7]. Let $\mathcal{E}$ denote the set of all possible operations for certain e. The possible operations are mixed through a softmax function. α is now related to a probability distribution over all operations, where each operation has its own weights.

$$\bar{e}^{(i,j)}(v^i) = \sum_{e \in \mathcal{E}} \frac{\exp(\alpha_e^{(i,j)})}{\sum_{e' \in \mathcal{E}} \exp(\alpha_{e'}^{(i,j)})} e^{(i,j)}(v^i) \tag{6}$$

Algorithm 1: Neural Architecture Search for Discriminator in Adversarial Training

```
Initialize $(\alpha, \theta, \phi)$
while not converged do
    Update $\alpha$ by descending $\nabla_\alpha \mathcal{J}_{val}(\alpha, \theta^*, \phi^*)$
    Update $(\theta, \phi)$ by descending $\nabla_{\theta,\phi} \mathcal{J}_{train}(\alpha, \theta, \phi)$
end
Derive the final architecture.
```

A discrete architecture can be decoded after α is learned. For each (i, j), only one operation is derived.

$$e^{(i,j)} = \operatorname*{argmax}_{e \in \mathcal{E}} \alpha_e^{(i,j)} \tag{7}$$

For node v^j, top-k $(0 < k \leq j)$ most likely operations are retained from all previous nodes $\{v^i | i < j\}$. The probability of $e^{(i,j)}$ is just the weight of this operation $(\frac{\exp(\alpha_e^{(i,j)})}{\sum_{e' \in \mathcal{E}} \exp(\alpha_{e'}^{(i,j)})})$ in Eq. 6. The probabilities over all possible i are calculated and the highest k nodes are chosen.

2.3 Optimization

With the continuous relaxation of α in Sec. 2.2, the architecture search can be achieved through gradient-based optimization. It is difficult to get a closed form solution to Eq. 3 and 4. Instead, we use an iterative approach to approximate the architecture gradient $\nabla_\alpha \mathcal{J}_{val}(\alpha, \theta^*, \phi^*)$, where

$$J_{val}(\alpha, \theta^*, \phi^*) = -\log g_{\alpha,\phi^*}(y) - \log(1 - g_{\alpha,\phi^*}(f_{\theta^*}(x)). \tag{8}$$

To optimize α, we approximate (θ^*, ϕ^*) by updating (θ, ϕ) with a single step on *train* [7]. So the optimization of α takes both (θ, ϕ) into consideration.

$$\begin{aligned} \theta^* &\approx \theta - \eta \nabla_\theta \mathcal{L}_{train}(\theta, \alpha) \\ \phi^* &\approx \phi - \eta \nabla_\phi \mathcal{L}_{train}(\phi, \alpha) \end{aligned} \tag{9}$$

Given α fixed, the optimization of (θ, ϕ) is is to minimize $\mathcal{J}_{train}(\alpha, \theta, \phi)$ by Eq. 1 and 2. We use the finite difference approximation in calculating the Hessian matrix in Eq. 9 suggested by [7]. The training procedure is outlined in Alg.1.

3 Experiments

3.1 Datasets

JSRT. JSRT is a classic dataset for lung nodule study released by the Japanese Society of Radiological Technology [13]. JSRT contains 247 chest X-ray images (CXRs) with pixel-wise ground truth labels of left lung, right lung and heart.

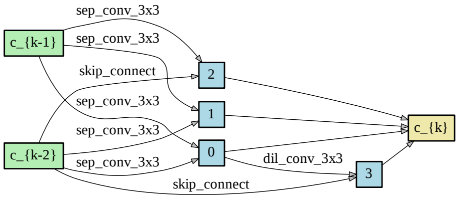

Fig. 3: An example for learned normal cell.

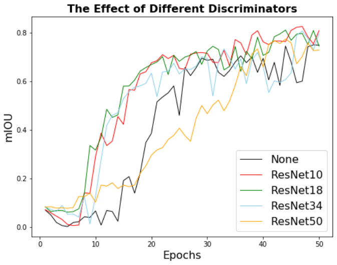

Fig. 4: *None* means FCN-32s without adversarial training.

Each CXR has a fixed resolution of 2048 × 2048 and was taken by the same machine under the same imaging protocols.

CX-SEG. CX-SEG is a new benchmark dataset for chest organ segmentation, which contains 259 CXRs with pixel-wise ground truth labels of left lung, right lung and heart. The annotation was conducted by 3 licensed radiologists. The CXRs were collected from multiple hospitals and various patient groups with different machines and imaging protocols. Compared with JSRT, CX-SEG shows large variation in terms of data sources, contrast, color and resolution, which can be used to test the robustness of the model. The CX-SEG is released for academic purpose.

3.2 Implementation

The proposed method is implemented in PyTorch based on the source code of [7]. We use the same search space for the set of possible operations $\mathcal{E}$ defined in [16, 7]. Two types of cells searched are *normal* cell and *reduction* cell. The reduction cell will downsample the resolution by using a stride 2 in the operations(edges) adjacent to the input nodes. For each cell, we set $B = 7$ and $k = 2$, which means 2 input nodes, 4 intermediate nodes, and 1 output node. An example of a learned normal cell is showed in Fig. 3. When designing the architecture of the discriminator, the stacked cells are the main body. In this study, we have four building blocks in the main body. In the first three building blocks, each normal cell is followed by a reduction cell, while there is only one normal cell in the last building block. We also add a pre-defined head and bottom to form a complete classification network. The head is one 3 × 3 convolutional layer and one 2 × 2 max pooling layer. The bottom is 1 fully-connected layer for binary classification.

We use a constant learning rate 10^{-4} and $\lambda_{adv} = 0.01$ in all experiments. Two classic segmentation networks are chosen as backbones, which are FCN-32s [8] and U-Net [12]. The images and ground truth labels are resized to 512 × 512. For each dataset, 50 images are randomly selected as the training data and the rest as the test data. We split the data this way to simulate a dataset with

Table 1: Results on CX-SEG

model	mIOU(%)
FCN-32s	75.27
FCN-32s + RS	72.69
FCN-32s + ResNet10	82.66
FCN-32s + NAS	82.43
U-Net	71.09
U-Net + RS	68.59
U-Net + ResNet10	79.32
U-Net + NAS	81.86

Table 2: Results on JSRT

model	mIOU(%)
FCN-32s	50.82
FCN-32s + RS	54.61
FCN-32s + ResNet10	85.86
FCN-32s + NAS	87.37
U-Net	43.95
U-Net + RS	44.85
U-Net + ResNet10	81.78
U-Net + NAS	83.78

limited supervision and to maximally highlight the influence of the different discriminators. There is no normalization or data augmentation on the training data. We use mean Intersection-Over-Union (mIOU) as the evaluation metric. The training data is further split into *train* and *val* by half. We use two GTX Titan X GPUs in this study. Each search takes at least 3 days. After the search is done, the learned model is trained on the whole training data for 50 epochs to give the final performance.

3.3 Evaluation

First, we use FCN-32s as the segmentation network and ResNet[6] as the discriminator. Intuitively, we illustrate the effect of the discriminator's architecture on the segmentation performance. The test results are presented in Fig. 4. Adversarial training with ResNet10 and ResNet18 outperform FCN-32s in both performance and convergence, while ResNet50 can only achieve a comparable result as non-adversarial training with poor convergence speed. ResNet10 is a variant of ResNet18 with only 4 residual blocks. Note, ResNets are already well-designed networks with guaranteed convergence. Poorly-designed discriminators cannot even guarantee convergence.

We use the proposed method to automatically design the discriminators for FCN-32s and U-Net in adversarial training. To fairly assess the proposed method, for each backbone network, we compare the NAS result with the segmentation network without adversarial training, adversarial training with the discriminator generated by *random search* (RS) and adversarial training with a well-designed discriminator. For the experiments of each backbone network, the segmentation networks share the same randomly initilized weights. For RS, we randomly initiate α 10 times to generate 10 different discriminators. We report the mean mIOU for 10 runs. We choose ResNet10 as the discriminator according to the test results in the last experiment. The results on CX-SEG and JSRT are presented in Tab.1 and Tab.2. Adversarial training improves the performance of the segmentation network by a large margin. The discriminator designed by NAS can achieve comparable even better results than the manually-designed discriminator. We conclude that the proposed method can be used to automatically design

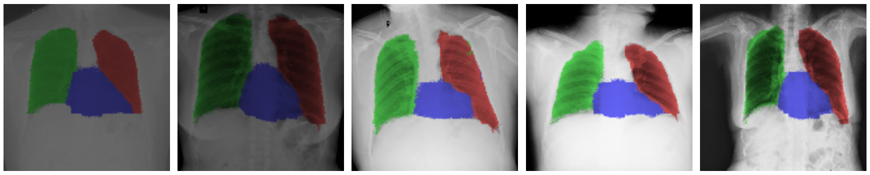

Fig. 5: Visualization of the chest organ segmentation on CX-SEG data.

the architecture of the discriminator in adversarial training with limited human intervention. Qualitative results on CX-SEG can be seen in Fig.5. We also want to point out that adversarial training often suffers from instability, which may also happen in our experiments. So the results maybe only the sub-optimal. We leave the theoretical discussion in future work.

4 Conclusions

In this paper, we present an approach to automatically design the architecture for the discriminator in adversarial training. We evaluate the proposed method with extensive experiments. This is the first study on neural architecture search in adversarial training while we admit that the proposed method requires further theoretical justification and still has a few limitations in both algorithmic and engineering perspective. In future work, we will optimize the memory storage mechanism and make the algorithm more efficient. It will also be interesting to incorporate the depth of the discriminator into the search space or jointly search for the architectures of both segmentation network and discriminator.

References

1. Baker, N., Lu, H., Erlikhman, G., Kellman, P.J.: Deep convolutional networks do not classify based on global object shape. PLoS Computational Biology 14(12), e1006613 (2018)
2. Dai, W., et al.: Scan: Structure correcting adversarial network for organ segmentation in chest x-rays. In: Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support, pp. 263–273. Springer (2018)
3. Dong, N., Kampffmeyer, M., Liang, X., Wang, Z., Dai, W., Xing, E.: Unsupervised domain adaptation for automatic estimation of cardiothoracic ratio. In: MICCAI. pp. 544–552. Springer (2018)
4. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: NIPS. pp. 2672–2680 (2014)
5. Han, Z., Wei, B., Mercado, A., Leung, S., Li, S.: Spine-gan: Semantic segmentation of multiple spinal structures. Medical Image Analysis 50, 23–35 (2018)
6. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016)

7. Liu, H., Simonyan, K., Yang, Y.: DARTS: Differentiable architecture search. In: ICLR (2019)
8. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR. pp. 3431–3440 (2015)
9. Luc, P., Couprie, C., Chintala, S., Verbeek, J.: Semantic segmentation using adversarial networks. In: NIPS *Adversarial Training Workshop* (2016)
10. Moeskops, P., Veta, M., et al.: Adversarial training and dilated convolutions for brain mri segmentation. In: Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support, pp. 56–64. Springer (2017)
11. Pham, H., Guan, M.Y., Zoph, B., Le, Q.V., Dean, J.: Efficient neural architecture search via parameter sharing. In: ICML. pp. 4092–4101 (2018)
12. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: MICCAI. pp. 234–241. Springer (2015)
13. Shiraishi, J., et al.: Development of a digital image database for chest radiographs with and without a lung nodule: receiver operating characteristic analysis of radiologists' detection of pulmonary nodules. American Journal of Roentgenology 174(1), 71–74 (2000)
14. Xie, L., Yuille, A.: Genetic cnn. In: ICCV. pp. 1379–1388 (2017)
15. Zoph, B., Le, Q.V.: Neural architecture search with reinforcement learning. In: ICLR (2017)
16. Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V.: Learning transferable architectures for scalable image recognition. In: CVPR. pp. 8697–8710 (2018)