Probabilistic Grammar-based Deep Neuroevolution

Pak-Kan Wong
The Chinese University of Hong Kong
Hong Kong
pkwong@cse.cuhk.edu.hk

Man-Leung Wong
Lingnan University
Hong Kong
mlwong@ln.edu.hk

Kwong-Sak Leung
The Chinese University of Hong Kong
Hong Kong
ksleung@cse.cuhk.edu.hk

ABSTRACT

Designing deep neural networks by human engineers can be challenging because there are various choices of deep neural network structures. We developed a deep neuroevolution system to automatically design deep neural network structures using deep neuroevolution. Our approach defines a set of structures using a probabilistic grammar and searches for best network structures using Probabilistic Model Building Genetic Programming. Our approach takes advantage of the probabilistic dependencies found among the structures of networks. The system was applied to tackle the problem of the physiological signal classification of abnormal heart rhythm. In the classification problem, our discovered model is more accurate than AlexNet. Our discovered model uses about 2% of the total amount of parameters of AlexNet.

CCS CONCEPTS

• **Software and its engineering** → **Genetic programming**; • **Computing methodologies** → *Neural networks*;

KEYWORDS

Estimation of Distribution Programming, Deep Neural Network

ACM Reference Format:
Pak-Kan Wong, Man-Leung Wong, and Kwong-Sak Leung. 2019. Probabilistic Grammar-based Deep Neuroevolution. In *Genetic and Evolutionary Computation Conference Companion (GECCO '19 Companion), July 13–17, 2019, Prague, Czech Republic.* ACM, New York, NY, USA, Article 4, 2 pages. https://doi.org/10.1145/3319619.3326778

1 INTRODUCTION

Recently, Deep Neural Network (DNN) is a powerful machine learning technique and attains outstanding performance in many medical applications, such as the identification of skin lesions [1]. Even a slight improvement of the performance for these detection systems implies an early diagnosis of diseases and save the people's lives. However, manually configuring of a DNN is complicated because the large search space of DNN covers many aspects, such as the topology of the network and the learning parameters. In this paper, we explore a physiological application of Deep Neuroevolution (DNE) to the automatically design of Convolutional Neural Network (CNN), which has not been reported before.

GECCO '19 Companion, July 13–17, 2019, Prague, Czech Republic

ACM ISBN 978-1-4503-6748-6/19/07.
https://doi.org/10.1145/3319619.3326778

In this paper, we propose a new procedure to evolve DNN structures from a set of convolutional layers, called modules, using Probabilistic Model Building Genetic Programming (PMBGP) approach. There are several advantages to this approach. First of all, it explicitly models the preference for combinations among different modules via probabilistic dependencies in the set of DNNs. PMBGP approach can automatically learn how to compose the highly dependent network modules. Secondly, since the DNN is represented in a grammar, new forms of regularities can be discovered to optimize the DNN structures. The system can decide what components to be inserted using the structural context, i.e. information about the location of a new component with respect to other existing components in a network structure. The context information can be nicely captured in Probabilistic Context-Sensitive Grammar (PCSG) while this is also the first study to apply PCSG and PMBGP on DNE.

2 DEEP NEURAL NETWORK STRUCTURE SEARCH

Our proposed Deep Neuroevolution (DNE) system can evolve DNNs in five steps (Figure 1):

(1) Deriving DNN structures encoded in parse trees according to the grammar;
(2) Translating the parse trees into Python code;
(3) Evaluating the performance of the network using data;
(4) Collecting samples from the parse trees of the set of good networks (by their ranking);
(5) Updating Bayesian networks in the grammar.

Steps 1 to 5 are repeated until it reaches the maximum number of generations. Our system is based on Grammar-based Genetic Programming with Bayesian Network (BGBGP) [8], and the evaluation step relies on PyTorch [7]. Stochastic gradient descent algorithm is adopted to optimize the weights in DNNs.

A DNN model contains a DNN structure of several DNN modules and a set of weights. A DNN topology is a specification describing a set of DNN structures of interest. We define eight network modules as the building blocks. An *input module* is a convolutional layer which has an incoming edge from the input data. An *output module* is a fully connected layer to transform its input to a class output in the one-hot encoding which can represent categorical variables as binary vectors. As for the remaining six modules, their incoming edges and outgoing edges connect to one of the eight network modules. A *BRC module* is composed of a batch normalization layer [3], a rectified linear layer [6], and a convolutional layer [5], which are connected in sequential order. An *aggregator module* is an addition function to combine multiple incoming edges into a single outgoing edge. Lastly, four *local features extraction* (LFE) modules can extract the local features from input at different scales in parallel. Let $LFE(m, n)$ denote a LFE module, where m and n are the size of the

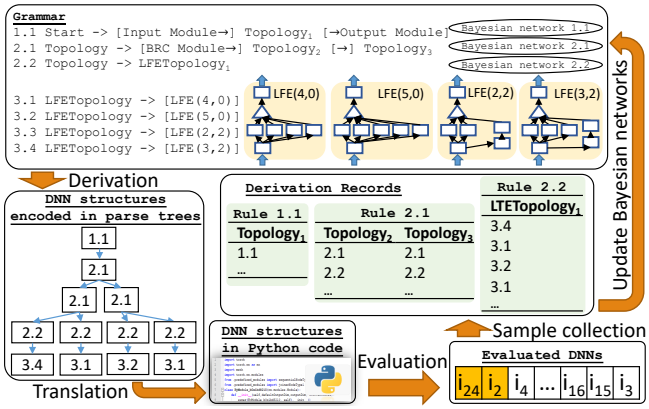

Figure 1: DNE system.

parallel part and the size of the sequential part respectively. Figure 1 shows the internal structure of LFE(4,0), LFE(5,0), LFE(2,2), and LFE(3,2). They are composed of several BRC modules (Rectangles) and aggregator modules (Triangles).

A *DNN topology grammar* encodes how DNN modules are assembled. It is a PCSG which comprises of a set of rules associated with a Bayesian network for each of them. The grammar in Figure 1 restricts the search space for the DNN structures. The 7 rules in the grammar are labelled from 1.1 to 3.4. Terminals are embraced by a pair of square brackets while other are non-terminals. Rule 1.1 defines the starting point and ending point of a network structure, i.e. an input module and an output module. Rule 2.1 says that a network topology *Topology* can be composed of two network topologies. The input to the compositions of network topologies is preprocessed by a BRC module. Rule 2.2 means that a LFE topology (*LFETopology*), which is defined in rules 3.1 to 3.4, is also a network topology. During the derivation of a parse tree, the structural context is provided by context variables and Bayesian networks can guide the search. The details of the derivation can be found in [8].

3 EVALUATION

A data set of the physiological measures from electrocardiography leads and pulse oximetry collected from four hospitals in the USA and Europe was downloaded [2]. Each record lasts 20 seconds long and contains 5000 features in total. Our goal is to detect if ventricular tachycardia occurs immediately after 20 seconds. There are 310 records. The records can be categorized into two classes. Patients who suffered from ventricular tachycardia, which contributes 29% of records. Another class of records were collected from patients who did not suffer from ventricular tachycardia. Stratified sampling was applied. The test, train, and validation sets contain 30%, 56%, and 14% of records respectively. In our experiment, we tested four other DNNs for comparison. Each of them was run for 50 runs. Then, the best model of each DNN was compared with our evolved network.

In our best model, the number of model parameters is around 0.064M using initial learning rate of 0.01. The network was trained for a maximum of 500 epochs. Four neural networks for comparison covered a different amount of parameters (from 0.031M to 24M) and

Table 1: Comparison with other DNN classifiers.

	DNE	AlexNet-A	AlexNet-B	Net-A	Net-B
Parameters	0.064M	24M	2.8M	0.080M	0.031M
Accuracy	**76%**	75%	75%	62%	56%

network structures. The results are shown in Table 1. DNE denotes the best model discovered using our method. AlexNet [4] is a well-known DNNs designed by experts in DNN. It is originally applied in image classification tasks. AlexNet, denoted by AlexNet-A, contains only eight layers: five convolutional layers and three fully connected layers. AlexNet-B is a simplified version of AlexNet-A and has 16 neurons in the fully connected layers, which is the same as that in our evolved network. DNE has slightly higher accuracy than that of AlexNet-A and AlexNet-B but the total number of parameters in DNE is only 2.2% of that in AlexNet-B.

There is no guarantee that the performance of the network will increase with the total number of parameters. To demonstrate this property, two neural networks were tested. Net-A is made of three fully connected layers and has 16 neurons. There are about 0.016M more parameters in Net-A than DNE does, but the performance of Net-A drops by 14%. Net-B connects a convolutional layer to a fully connected layer such that the number of parameters is only half of that in DNE. The performance drops by 20%. This suggests that neuroevolution on the DNN structure is vital to achieve high accuracy using a small number of parameters.

4 CONCLUSIONS

We proposed a deep neuroevolution approach using a probabilistic grammar-based method. It can not only improve but learn DNN structures. It is evident that DNNs attaining comparable predictive performance can be evolved while deploying our DNNs requires much less computing resources. It will help experts to adopt DNN to analyze complex data sets. In the future, we will investigate if our system can learn novel DNN structures for complex data sets, such as those with many modalities.

ACKNOWLEDGMENTS

This research is supported by the Institute of Future Cities of The Chinese University of Hong Kong.

REFERENCES

[1] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542(7639):115, 2017.
[2] A. L. Goldberger, L. A. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley. Physiobank, physiotoolkit, and physionet. *Circulation*, 101(23):e215–e220, 2000.
[3] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proc. of the 32th ICML*, pages 448–456, 2015.
[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. of the NIPS 2012*, pages 1097–1105, 2012.
[5] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.
[6] V. Nair and G. E. Hinton. Rectified linear units improve restricted Boltzmann machines. In *Proc. of the 27th ICML*, pages 807–814, 2010.
[7] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.
[8] P.-K. Wong, L.-Y. Lo, M.-L. Wong, and K.-S. Leung. Grammar-based genetic programming with Bayesian network. In *Proc. of IEEE CEC 2014*, pages 739–746. IEEE, 2014.