Evolving Convolutional Neural Networks through Grammatical Evolution

Ricardo H. R. Lima
Federal University os Paraná
Curitiba, Paraná
rhrlima@inf.ufpr.br

Aurora T. R. Pozo
Federal University of Paraná
Curitiba, Paraná
aurora@inf.ufpr.br

ABSTRACT

The use of Convolutional Neural Networks (CNNs) have proven to be a solid approach often used to solve many Machine Learning problems, such as image classification and natural language processing tasks. The manual design of CNNs is a hard task, due to the high number of configurable parameters and possible configurations. Recent studies around the automatic design of CNNs have shown positive results. In this work, we propose to explore the design of CNN architectures through the use of Grammatical Evolution (GE), where a BNF grammar is used to define the CNN components and structural rules. Experiments were performed using the MNIST and CIFAR-10 datasets. The obtained results show that the presented approach achieved competitive results and maintaining relatively small architectures when compared with similar state-of-the-art approaches.

CCS CONCEPTS

• **Computing methodologies** → **Genetic algorithms**; Neural networks;

KEYWORDS

grammatical evolution, convolutional neural networks, automatic design

ACM Reference Format:
Ricardo H. R. Lima and Aurora T. R. Pozo. 2019. Evolving Convolutional Neural Networks through Grammatical Evolution. In *Proceedings of Genetic and Evolutionary Computation Conference Companion (GECCO '19 Companion)*. ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3319619.3322058

1 INTRODUCTION

Convolutional Neural Networks (CNNs) are a powerful approach usually applied to computer vision tasks [9, 10]. The use of CNNs is one of the reasons on why Deep Learning (DL) is a very popular area of study in recent years [8, 11].

Designing CNNs is a complex task, due to the high number of possible configurations, including selecting layers and layer parameters. Thus, the use of techniques that automatically design CNN architectures have become more attractive. The construction of CNN architectures can be expressed as a hyperparameter optimization problem [1, 2], where layers and layer parameters are all considered variables that can be optimized using search techniques, such as evolutionary algorithms [7, 9, 10].

In this work, we deal with the design of CNNs using Grammatical Evolution (GE). It is a flexible approach that uses a context-free grammar to describe how to build CNN architectures. The models are executed to compute the fitness. An empirical study is made to evaluate the approach on image classification tasks using the MNIST [6] and CIFAR-10 [5] datasets, comparing the results with other state-of-the-art approaches.

2 DESIGNING CNN ARCHITECTURES WITH GRAMMARS

The approach we propose is very straight forward. Through the search engine (GA), a population of solutions iteratively selects, modifies, evaluates and replaces solutions, in order to improve the quality of the population. At the evaluation step, first the solutions have to be translated into an actual CNN. Then, the model is executed on the training set and the fitness is computed based on the performance on the evaluation set. Figure 1 summarizes the process.

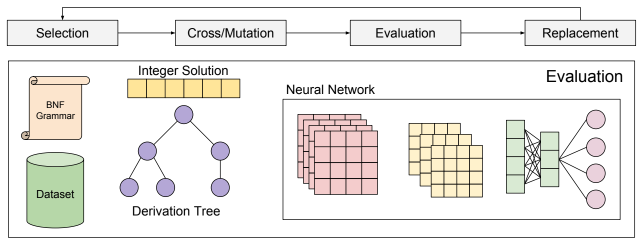

Figure 1: Representation of the proposed approach. While the evolutionary algorithm is very simple, the evaluation step is where the solutions are translated from integer vectors to CNN architectures using the BNF grammar and then executed to compute the fitness.

The proposed Grammar (Figure 2) was built to offer enough flexibility in the design of either a small network (one convolution and one dense layer, for example) or even bigger networks with many layers. In this work, we focused only on sequential architectures, where the models are formed by stacked layers, each with its own parameters.

⟨*cnn*⟩ ::= ⟨*conv*⟩ ⟨*c_layer*⟩ ⟨*d_layer*⟩ ⟨*dense*⟩

⟨*c_layer*⟩ ::= ⟨*c_layer*⟩ ⟨*c_layer*⟩ | ⟨*c_node*⟩ | '&'

⟨*d_layer*⟩ ::= ⟨*d_node*⟩ ⟨*d_node*⟩ | ⟨*d_node*⟩ | '&'

⟨*c_node*⟩ ::= ⟨*conv*⟩ | ⟨*maxpool*⟩ | ⟨*avgpool*⟩

⟨*d_node*⟩ ::= ⟨*dense*⟩ | ⟨*dropout*⟩

⟨*conv*⟩ ::= '`Conv2D`' ⟨*filters*⟩ ⟨*k_size*⟩ ⟨*activation*⟩

⟨*dense*⟩ ::= '`Dense`' ⟨*units*⟩

⟨*dropout*⟩ ::= '`Dropout`' ⟨*rate*⟩

⟨*maxpool*⟩ ::= '`MaxPooling2D`' ⟨*p_size*⟩ ⟨*padding*⟩

⟨*avgpool*⟩ ::= '`AveragePooling2D`' ⟨*p_size*⟩ ⟨*padding*⟩

⟨*activation*⟩ ::= '`relu`' | '`selu`' | '`elu`'
| '`tanh`' | '`sigmoid`' | '`linear`'

⟨*padding*⟩ ::= '`valid`' | '`same`'

⟨*filters*⟩ ::= '`16`' | '`32`' | '`64`' | '`128`'

⟨*k_size*⟩ ::= '`(3, 3)`' | '`(5, 5)`' | '`(7, 7)`'

⟨*p_size*⟩ ::= '`(2, 2)`' | '`(4, 4)`'| '`(6, 6)`'

⟨*units*⟩ ::= '`32`' | '`64`' | '`128`' | '`256`'

⟨*rate*⟩ ::= '`[0.0, 1.0]`'

Figure 2: Proposed grammar to generate CNN architectures.

3 EXPERIMENTS

This experimental step will be used to evaluate our approach, here called GE-CNN. To do this, 5 independent runs were performed, with both 600 and 1000 max evaluations. During the evolution, each model is trained for 50 epochs using the training set, and then executed on the validation set to calculate the fitness. The optimizer used was Adam [4]. For the evolutionary algorithm, the genetic operators we used are: random selection, one-point crossover, point mutation, prune and, duplication. At the end, the best design found is executed again on the training set, this time for 500 epochs, and then tested 30 times using the test set. The experiments were performed using the MNIST [6] and CIFAR-10 [5] datasets.

The best accuracy reported for the MNIST experiment (Table 1) is 98, 95% from the LDANet-2 [3], considering only the mean value. Our approach was able to achieve an accuracy of 99, 03% on the best case, and 98, 78% on average among 30 runs. Similarly, for the CIFAR-10 (Table 2), GE-CNN achieved an accuracy of 76, 34% on the best case and 72, 84% on average, a very similar result compared to the CGP-CNN (ResSet) [9] approach, that achieved an accuracy of 76, 53% on average. The generated models have a relatively small design with 5 and 7 layers respectively.

Table 1: Experiments with MNIST dataset

Model	Acc
ScatNET-2	98,73
RandNet-2	98,75
LDANet-2	**98,95**
EUDNN/AE	98,78
GE-CNN (mean)	98,78
GE-CNN (best)	**99,03**

Table 2: Experiments with CIFAR-10 dataset

Model	Acc
VGG	75,89
ResNet	75,90
CGP-CNN(ConvSet)	76,52
CGP-CNN(ResSet)	**76,53**
GE-CNN (mean)	72,84
GE-CNN (best)	76,34

4 CONCLUSION

In this study, we proposed the use of Grammatical Evolution, that uses a context-free grammar to define the architecture layers and parameters to build CNNs. According to the experiments, our approach GE-CNN presented a simple and flexible approach to design CNN architectures. The evolutionary algorithm from GE can develop the architectures over time, maintaining the networks with a relatively small design, achieving competitive results. Still, there are many unexplored topics regarding the automatic design of deep neural networks. Expanding the representation power of the grammar by using more complex types of layers and network structures. Also, testing different evolutionary algorithms as the search engine, exploring different encoding and operators, as well as finding new ways to decrease the complexity and cost related to the evaluation step.

REFERENCES

[1] James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research* 13, Feb (2012), 281–305.
[2] James Bergstra, Daniel Yamins, and David Daniel Cox. 2013. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. (2013).
[3] Tsung-Han Chan, Kui Jia, Shenghua Gao, Jiwen Lu, Zinan Zeng, and Yi Ma. 2015. PCANet: A simple deep learning baseline for image classification? *IEEE Transactions on Image Processing* 24, 12 (2015), 5017–5032.
[4] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
[5] Alex Krizhevsky and Geoffrey Hinton. 2009. *Learning multiple layers of features from tiny images*. Technical Report. Citeseer.
[6] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.
[7] Ilya Loshchilov and Frank Hutter. 2016. CMA-ES for hyperparameter optimization of deep neural networks. *arXiv preprint arXiv:1604.07269* (2016).
[8] Risto Miikkulainen, Jason Liang, Elliot Meyerson, Aditya Rawal, Daniel Fink, Olivier Francon, Bala Raju, Hormoz Shahrzad, Arshak Navruzyan, Nigel Duffy, et al. 2019. Evolving deep neural networks. In *Artificial Intelligence in the Age of Neural Networks and Brain Computing*. Elsevier, 293–312.
[9] Masanori Suganuma, Shinichi Shirakawa, and Tomoharu Nagao. 2017. A genetic programming approach to designing convolutional neural network architectures. In *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 497–504.
[10] Yanan Sun, Bing Xue, and Mengjie Zhang. 2017. Evolving deep convolutional neural networks for image classification. *arXiv preprint arXiv:1710.10741* (2017).
[11] Yanan Sun, Gary G Yen, and Zhang Yi. 2018. Evolving Unsupervised Deep Neural Networks for Learning Meaningful Representations. *IEEE Transactions on Evolutionary Computation* (2018).