# Logical Explanations for Neural Networks Assisted by Neuroevolution

Abhista Partal Balasubramaniam
Technical University of Denmark
abhistap@gmail.com

Nina Gierasimczuk
Technical University of Denmark
nigi@dtu.dk

We explore a novel combination of methods to provide explanations for the outputs of neural networks using *logic explained networks* (LEN, [2]) combined with the genetic algorithm *neuroevolution of augmenting topologies* (NEAT, [5]). The new algorithms' performance and explanations are benchmarked in the well-known *cart-pole balancing* control problem. This contributes to the field of explainable AI, as it provides a bridge between symbolic and sub-symbolic AI.

***Keywords—*** Explainable Artificial Intelligence (XAI), Neural Networks, Fuzzy Logic, Neuroevolution of Augmenting Topologies (NEAT), Logic Explained Networks (LEN)

## 1 Introduction

Genetic algorithms are mainly used to solve reinforcement learning problems. In particular, *neuroevolution of augmenting topologies* (NEAT) is good at training neural networks by evolving the network architecture. NEAT increases the complexity of the neural network only if its current, simpler version is unable to find a solution. Since the neural network is evolving incrementally through crossover and mutation, it can be argued that the NEAT's architecture is, in some sense, minimal. On the other hand, *logic explained networks* (LENs) force the neural network to follow logical constraints expressed in propositional logic by modifying the loss function. The LEN booleanizes inputs to form *concepts*, which act like literals in a propositional sentence (so that the explanations are propositional logic formulas). This restricts the input and output spaces to only boolean values, which in turn reduces the power of neural networks. After training, the LEN goes through a step called pruning, i.e., reducing the length of the explanations given by the final neural network with minimum loss in expressivity (the ability to approximate complex functions) and explainability. Pruning is achieved by removing concepts that do not significantly contribute the final decisions. This is done by looking at the weights associated with the neural network's input concepts. Therefore, the pruning step reduces the accuracy of the network, but increases the readability of the explanations provided by the network.

We combine the two algorithms by replacing the pruning step of LEN by the minimal architecture supplied by NEAT. The LEN algorithm will restrict the input and output spaces of the neural network in order to provide propositional logic explanations. To keep the expressivity of the neural network, we use fuzzy logic, as it is capable of handling non-boolean logical values. Therefore, the new concepts for the combined algorithm have fuzzy variables, which in turn provide fuzzy logic explanations as the output of the LEN algorithm.

## 2 Methodology

There are three key modifications to the structure of the NEAT algorithm to incorporate the LEN component, see Figure 1: 1) a pre-processing block is added to each neural network in a *generation*, which takes care of booleanization or mapping of the inputs to fuzzy variables; 2) a post-processing block is added to each neural network in a generation, which takes care of the extraction of explanations provided by the neural network while training; and finally 3) the *fitness function* of the neural network is modified to incorporate the bi-directional loss function used in the LEN algorithm:

$$L_{\leftrightarrow}(\bar{y}_i, f, C) = \sum_{c \in C} \bar{y}_i(c)\ \log(f_i(c)) + (1 - \bar{y}_i(c))\ \log(1 - (f_i(c))) \tag{1}$$

where, $L_{\leftrightarrow}$ is called the bi-implication loss function; $\bar{y}_i$ is the actual $i^{th}$ output concept or label; $f_i$ is the output of LEN; $C$ is the set of all input concepts, and $r$ is the total number of input concepts. The new fitness function already accounts for non-binary outputs and hence the output of the neural network can be a fuzzy variable, which then undergoes post-processing to convert the fuzzy variable output of the neural network into the required output format. The new fitness function requires labeled data, hence making this a supervised learning algorithm. In this work, three different versions of the algorithm were implemented: (V1) NEAT and LEN, (V2) NEAT and LEN with *partial booleanization*, and (V3) NEAT and LEN with fuzzy logic [6].[1] V2 performs booleanization of the input space without loss of information, which is achieved by allowing the concepts to be real numbers instead of boolean values. The combined algorithm provides an output explanation for each decision taken by the neural network

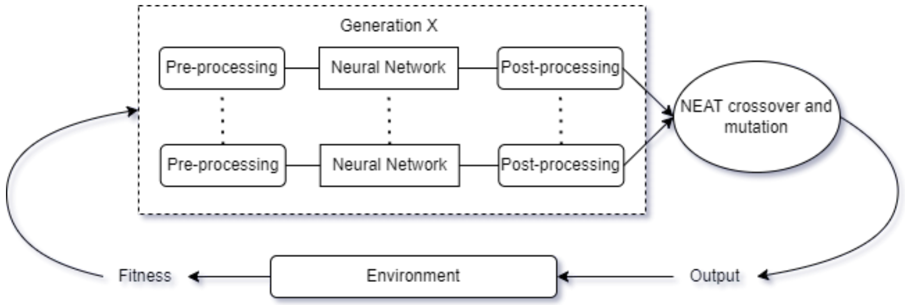


Figure 1: Modified structure of the NEAT algorithm.

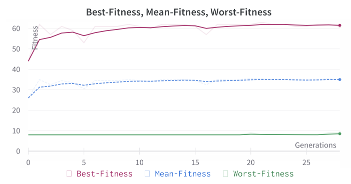
during and after training as a conjunction of concepts. The partial explanations (which can be seen as predictions) are of the form: $c_1 \wedge c_2 \wedge \ldots \wedge c_n \rightarrow D$, where $c_1, \ldots, c_n$ are the input concepts and $D$ is the decision. We can write the full, bi-directional explanations in DNF (Disjunctive Normal Form) form as follows: $D \leftrightarrow V^1 \vee V^2 \vee \ldots \vee V^m$, where the right-hand side represents all the vectors that lead to the decision $D$ made by the neural network, and for each $i \in \{1, \ldots, m\}$, $V_i$ is a shorthand notation for $c_1^i \wedge c_2^i \wedge \ldots \wedge c_n^i$. Since the loss function in Equation (1) optimizes for the above bi-implication formula, the explanations are provided are in that form. From the bi-implicatiomn we can then retrieve the implications. Therefore, each decision taken by the neural network can be given an explanation in that form.

## 3 Results

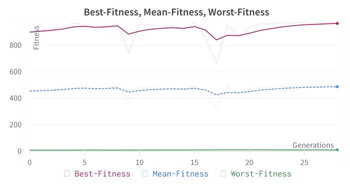
Each version of the combined algorithm was benchmarked in the cart-pole balancing environment [4, 3] using the metrics of: fitness, network topology, explanation accuracy, number of species, genome length, and explanation length. For simplicity, only the first three are presented in this abstract.

[1]The code can be found in the Github repository [1].

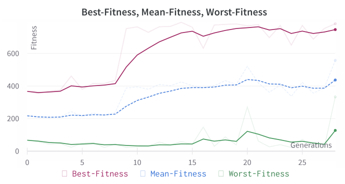
The **fitness** of a neural network in NEAT is the ability of the neural network to solve a given problem relative to the other neural networks evolved during the run of the algorithm. The fitness range chosen for this problem was between 0 to 1000, and shown in Figure 2.[2] The fitness of V1 indicates that the algorithm was unable to solve the cart-pole problem and provide explanations at the same time. V2 performs significantly better than V1: partial booleanization retains the information provided by the input, and so this version is able to solve the cartpole problem easily. V3 has good fitness, while simultaneously achieving decent explanation accuracy in the network's logical explanations.



(a) Fitness for V1.



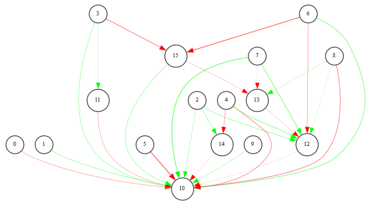
(b) Fitness for V2.

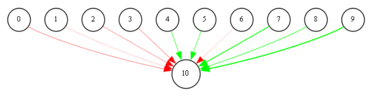


(c) Fitness for V3

Figure 2: Fitness evolution for each version of the implementation.

Figure 3 shows that the **neural network topologies** generated by the algorithm are simple for versions V2 and V3, whereas V1 (which struggles to solve the cart-pole balancing problem) is exploring more complex architectures.


(a) Network structure for V1.


(b) Network structure for V2.

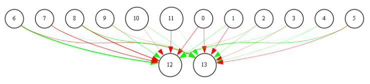

(c) Network structure for V3.

Figure 3: Network structure for each version of the implementation; the red, green, and dotted connections represent negative weight, positive weight, and disabled connections, respectively.

**Explanation accuracy** refers to the accuracy of the explanations provided by the trained neural network, i.e., the ability of the neural network to provide correct logical explanations for the decisions taken by the trained neural network. Therefore, an explanation accuracy of 70% tells us that if this neural network provides 10 logical explanations in the implicative form, then it can be expected that 7 of them are correct. We cannot tell which of the explanations are incorrect because the full explanations are in DNF. Hence, the explanation accuracy only talks about the explanatory power of the algorithm and not the decision-making ability of the neural network. From Table 1 it can be seen that only in the fuzzy logic implementation we can test the provided explanations, because in V1 the neural network generated was unable to balance the pole long enough to learn its explanations (hence, the below 50% accuracy in V1 training), and in V2 partial booleanized inputs do not follow the rules of propositional logic. For V3, since the neural network provides explanations in fuzzy logic, a fuzzy logic system was used to evaluate the explanations by using the explanations as inference rules. The explanation accuracy of 62.02% was

[2]Note that the $y$-axis limits are [0, 100], [0,1000], and [0, 800] in graphs (a), (b), and (c), respectively. The maximum fitness of the three versions is 65, 963, and 694, respectively.

| - | Explanation type | Explanation accuracy (25 runs) | |
|---|---|---|---|
| | | Training accuracy | Testing accuracy with logical system |
| V1 | Propositional logic | 12.21% | - |
| V2 | Propositional logic | 90.91% | - |
| V3 | Fuzzy logic | 56.71% | 62.02% |

Table 1: Explanation accuracy for each version.

obtained in a Monte Carlo simulation with 25 random runs. Here, the low explanation accuracy can be attributed to a ‘collapse’ in the explanation algorithm: the provided explanations are always the same and correct 50% of the time. A method to reduce the risk of the explanation algorithm collapsing can drastically increase the testing accuracy during the Monte Carlo simulation run.

## 4 Discussion and conclusion

The results obtained indicate that the provided algorithm has a lot of potential, as it can be used to explain the decisions made by a neural network. The low explanation accuracy could be improved by fine-tuning the parameters for the NEAT algorithm and, also, by providing the NEAT algorithm with more domain-specific information. The explanation accuracy in V1 is subpar because the neural network is unable to develop a strategy to solve the cartpole problem. This reflects an intuitive requirement that a neural network should be able to develop a strategy to solve the problem before it attempts to provide explanations. The simple network topologies produced by V2 and V3 indicate that the combined algorithm is able to solve the cart-pole problem without adding hidden nodes which is to be expected from prior experiments with NEAT.

To sum up, we have shown that NEAT and LEN can be combined to provide logical explanations. The new algorithm can use fuzzy logic explanations to increase the expressivity of the neural network. It thus provides another method at the interface between symbolic and sub-symbolic methods in AI.

## References

[1] A.P. Balasubramaniam (2023): *Logic for Neural Networks Assisted by Neuroevolution.* Master Thesis at DTU Compute, GitHub repository: `https://github.com/AbhistaPB/Thesis.git`.

[2] G. Ciravegna, P. Barbiero, F. Giannini, M. Gori, P. Liò, M. Maggini & S. Melacci (2023): *Logic Explained Networks.* *Artificial Intelligence* 314, p. 103822, doi:https://doi.org/10.1016/j.artint.2022.103822.

[3] L.P. Kaelbling, M.L. Littman & A.W. Moore (1996): *Reinforcement learning: A survey.* *Journal of artificial intelligence research* 4, pp. 237–285.

[4] S. Kumar (2020): *Balancing a CartPole System with Reinforcement Learning - A Tutorial.* *CoRR* abs/2006.04938. arXiv:2006.04938.

[5] K.O. Stanley & R. Miikkulainen (2002): *Evolving Neural Networks through Augmenting Topologies.* *Evolutionary Computation* 10(2), pp. 99–127, doi:10.1162/106365602320169811.

[6] L. Zadeh (1988): *Fuzzy logic.* *Computer* 21(4), pp. 83–93, doi:10.1109/2.53.