Adding Interpretability to Neural Knowledge DNA

Junjie Xiao [a], Tao Liu [a], Haoxi Zhang [a b], Edward Szczerbicki [c]

[a] School of Cybersecurity, Chengdu University of Information Technology, Chengdu, China; [b] Advanced Cryptography and System Security Key Laboratory of Sichuan Province, Chengdu, China; [c] Faculty of Management and Economics, Gdansk University of Technology, Gdansk, Poland

Abstract. This paper proposes a novel approach that adds the interpretability to Neural Knowledge DNA (NK-DNA) via generating a decision tree. The NK-DNA is a promising knowledge representation approach for acquiring, storing, sharing, and reusing knowledge among machines and computing systems. We introduce the decision tree-based generative method for knowledge extraction and representation to make the NK-DNA more explainable. We examine our approach through an initial case study. The experiment results show that the proposed method can transform the implicit knowledge stored in the NK-DNA into explicitly represented decision trees bringing fair interpretability to neural network-based intelligent systems.

Address correspondence to Haoxi Zhang, School of Cybersecurity, Chengdu University of Information Technology, No. 24 Block 1, Xuefu Road, Chengdu, China, 610225. E-mail: haoxi@cuit.edu.cn

Keywords: Interpretable AI, Neural Knowledge DNA, decision trees, deep reinforcement learning.

INTRODUCTION

The Neural knowledge DNA (NK-DNA) is a promising knowledge representation method for AI systems. Together with deep neural networks (DNNs) and reinforcement learning, NK-DNA is able to catch knowledge. It acquires knowledge in optimizing an agent's policy for better future rewards by tuning parameters of multi-layered neural networks. However, the DNNs work as a black box leading to the inability to interpret, which hugely discourages applying the NK-DNA to many important domains such as healthcare, aircraft, and finance, where the decision procedures are vital. Thus, any decisions made must be reasonable and interpretable.

To address this issue of the NK-DNA, in this paper, we propose a decision tree-based method collaborating with the NK-DNA to distil the knowledge learned by NK-DNA and transform it into a tree-like structure, allowing people to infer and understand the decision procedure and criteria of any decision made by the AI system.

The rest of the paper is organized as follows: Section 2 (Interpretable AI) explains the importance of interpretability in AI. In section 3 (Related Work), deep reinforcement learning & the NK-DNA are introduced. The method, experiments, and results are illustrated in Section 4 (Interpretable Module for NK-DNA). Finally, Section 5 (Conclusion) concludes the paper.

INTERPRETABLE AI

Artificial intelligence (AI) has made remarkable achievements in research and industrial areas, especially since the success of deep learning (DL). DL is a methodology mainly based on multi-layer artificial neural networks (NNs) to simulate the behaviour of the human brain and learn from large amounts of data. Although ANNs and related technologies have recently shown outstanding performance in many tasks, they are lack of interpretability which can lead to frailty (C. J. Kelly and A. Karthikesalingam et al. 2019). In some domains, there is no tolerance for any failure. For instance, early detection of the disease is usually critical to curing patients or stopping the disease from progressing to a more severe stage.

Consequently, the interpretability of the AI algorithm has become an urgent problem (Erico Tjoa and Cuntai Guan. 2020): who is responsible if there is something wrong? Can we explain why things go wrong? If everything goes well, do we know why and how to leverage them further? Many papers have suggested different methods and frameworks for achieving interpretability, and explainable artificial intelligence (XAI) is now a hot topic in AI research. Moreover, the introduction of interpretability evaluation criteria (such as causality, availability, and reliability) enables the AI researchers and engineers to track the logic and decision-making procedures of the algorithms and provide guidance for further improvement and development of AI systems (S. Tonekaboni and S. Joshi et al. 2019).

DEEP REINFORCEMENT LEARNING

Reinforcement learning (RL) is a branch of machine learning, which focuses on using the experience gained through interaction with the environment and assessing feedback to improve the system's decision-making ability (T.P. Lillicrap. 2015). RL algorithm is mainly inspired by the perception of the human decision-making process (R.S. Sutton and A.G. Barto. 1998). In human decision-making, humans learn how to respond to observed action outcomes by using reward signals in the brain. To mimic the human decision-making behaviour, the RL learns to perform well by feedbacks from the environment. Additionally, due to the continuous increase of rich data, exciting progress has been made in the theory and practice of reinforcement learning in recent years, including the development of primary technology fields, such as empirical methods, exploration, planning and generalization, which hugely improve the applicability to practical problems (M.L. Littman. 2015).

RL can be expressed as the interaction between learners (i.e. decision-makers) and the environment that provides evaluation results to the learners. The environment is usually understood as a Markov decision process (M. Puterman. 1994). The Markov decision process consists of a set of actions A (decisions that decision-makers can choose) and states S (situations where decisions can be made). The number of these actions and states may be limited, but in some reinforcement learning applications, such as performing physical tasks, space with continuous actions and states is usually more valuable. The function p (s' | s, a) defines the probability of transforming the state from s to s' by taking action a (M.L. Littman. 2015).

Reward function R (s, a) and discount variable $\gamma \in [0,1]$ are used to express the performance of the decision-making agent: in each step, the agent selects an action, the environment feedbacks a reward and converts it into the next state. The goal of the agent is to maximize the sum discount expected reward from the environment. In other words, the agent is finding a policy $\pi^*(a_t \mid s_t; \theta)$ mapping the states to the action of generating the reward sequence r_0, r_1, r_2, r_3, ...r_t such that $E_{r0, r1, \ldots}$ $[r_0 + \gamma * r_1 + \gamma^2 * r_2 + \gamma^3 * r_3 + \ldots + \gamma^t * r_t]$ as large as possible. The Bellman equation (R. Bellman. 1957) of the optimal state action-value function Q* captures the relationship between the cumulative discount expected reward and environmental interactions (state, action, reward, state, action, reward, etc.). The solution of the Bellman equation enables the agent to optimize its behaviour by calculating $\pi^*(s)$ = argmax Q*(s, a). The expected cumulative discounted reward for the policy that takes action a from state s and then behaving optimally thenceforth is the immediate the reward received, and the expected discounted value of the cumulative discounted expected reward from the resulting state s' given that the best action is chosen (M. Puterman. 1994).

Deep Reinforcement Learning (DRL) is a method that combines DNNs with RL to resolve the learning environment and obtain the best control strategy. DNNs may be used to derive a straight approximation of the control strategy: $a = \pi(s)$, from examples of data points (s_i, a_i) resulting in other control processes. In the Neural Knowledge DNA, the DRL is used to capture implicit knowledge for the agent.

NEURAL KNOWLEDGE DNA

The Neural Knowledge DNA (NK-DNA) was proposed by Zhang et al. (2017) to support acquiring, storing, and sharing knowledge in different artificial intelligence systems that use neural networks as the main power of its intelligence. They modify the ideas underlying the success of deep learning (LeCun et al. 2015) to the extent of knowledge representation.

The NK-DNA is organized in a similar form to the DNA (Sinden 1994): consisting of four critical sections. Our NK-DNA borrows the idea of DNA to store knowledge. As the DNA produces phenotypes, the NK-DNA carries knowledge and information through its four essential elements: States, Actions, Experiences, and Networks (see Figure 1).

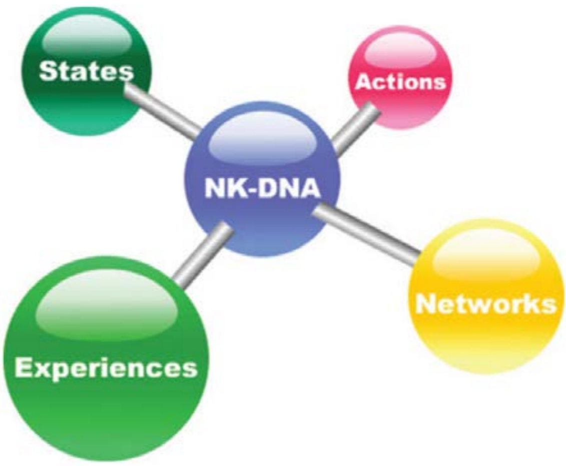

Fig. 1. The structure of the NK-DNA.

The NK-DNA's combination of four essential elements is conceived to carry detailed information about the decisions: States are conditions in which a decision or an action can be made or done. Actions that are used to express the decisions or motions of the domain can choose. Experience is the historical operation data of the system, with feedback from the results. The Networks store the details of the artificial neural networks for training and utilizing that knowledge, such as the structure of networks, weights, bias and the in-depth learning framework used.

Generally speaking, after training in a deep learning system, knowledge is obtained as a model. The model stores information as the weights and biases of connections among neurons of neural networks, as well as a neural network hierarchy in detail. Once the neural network is trained, it will directly give results by calculating through its network layers after input.

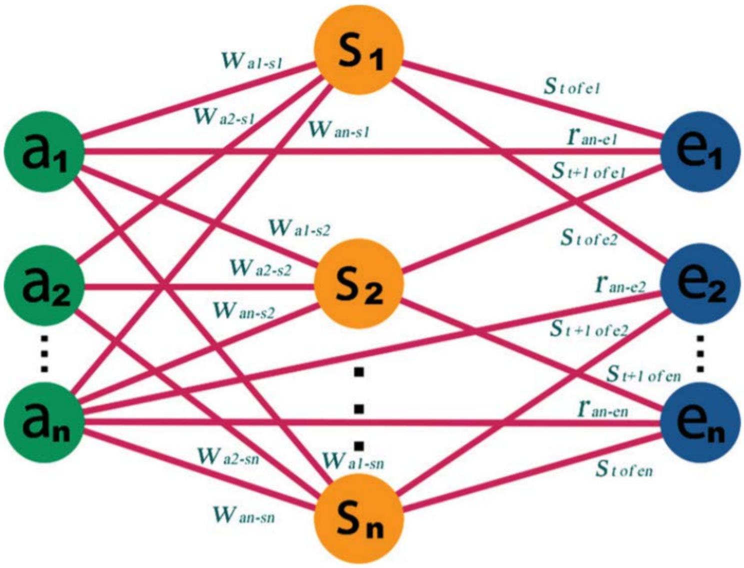

Fig. 2. The NK-DNA-carried knowledge.

In NK-DNA, the Networks are used to carry the relationship between actions and states: as shown in Figure 2, each state (expressed as s_1, s_2... s_n) can be linked with a set of actions (defined as a_1, a_2... a_n). If an action is associated with a state, the related action is available in that state; In other words, if it is in this state, the agent can select the action to perform. The trained neural network provides knowledge about which action is the best choice for a particular state. The state here is the input, which can be the original sensory data or the data representing the agent's current state.

Another essential feature of this method is that NK-DNA uses past decision-making experiences to gather and expand the intelligence to support future decision-making. Generally, agents transform from one state to another during their operation, make decisions (select actions) in each state, and receive feedback from their operations. These states, actions, feedbacks and transitions compose the so-called "experience" (Sanin& Szczerbicki. 2006).

INTERPRETABLE MODULE FOR NK-DNA

A. OVERVIEW

Although the NK-DNA can allow the intelligent system to acquire, store, share, and reuse the knowledge through the daily operation of the domain, the NK-DNA's multi-layer neural networks, which is used as a function approximator to catch and store knowledge, works as a black box and is uninterpretable. Therefore, we add a highly interpretable decision tree-based module to extract and represent the knowledge from the neural network of the NK-DNA (see Figure 3) to improve the interpretability of NK-DNA.

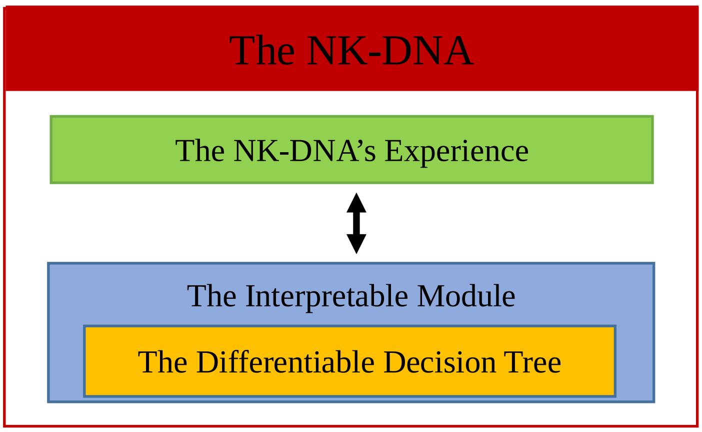

Fig. 3. NK-DNA with the added Interpretable Module.

B. THE MODULE DETAILS

The decision tree (DT) is an efficient and reliable decision-making technology, which uses a tree-like structure to store a simple representation of gathered knowledge (i.e. patterns within data) and control decision-making (Shah and Gopal, 2010). And DTs are regarded as the practicable technology for interpretable and transparent ML (Andrew Silva and Taylor Killian et al. 2020).

However, the standard decision trees are barely used in online training tasks of RL. The non-differentiable feature of the decision tree causes it cannot be updated via gradient descent. Suárez and Lutsko (1999) provide the first differentiable decision tree (DDT) models. They change the edge of the standard decision tree by using the sigmoid activation function (Eq. 1)

$$\mu(x) = \frac{1}{1+e^{-(\alpha_\eta(\beta_\eta^T x-\emptyset_\eta))}} \tag{1}$$

to give a smooth transition between 0 and 1, making the edge variable differentiable: it uses a linear features *x* weighted by β_η compared to a bias ϕ_η, and augmented by a steepness parameter α_η (Suárez and Lutsko, 1999). The tree is trained via gradient descent for tuning parameters β_η, ϕ_η, and α_η across nodes η.

Although their method has been applied to off-line and supervised learning, it still has not been applied to online RL. The reason behind this is that there are two key drawbacks in the method: First, the original operation $\beta_\eta^T x$ considers a linear combination of the features at each node to compare with ϕ_η, rather than a single feature comparison. Second, using the sigmoid activation function means a smooth transition between True and False evaluation of nodes, rather than discrete decisions. Andrew Silva and Taylor Killian et al. (2020) address these two issues by employing an $\arg max_j \left(\beta_\eta^j\right)$ to convert the differentiable tree into a truly discrete tree and obtain the index of the feature normalizing the node will use. They also divide ϕ_η by the node's weight β_η^j, normalizing the value for comparison against the raw input feature x_j. Each node then compares a single raw input feature to a single ϕ_η. Furthermore, they also use the $\arg max_j \left(\beta_\eta^j\right)$ for each leaf as well as decision node to obtain a final interpretable decision tree with discrete nodes.

Our method applies the optimized DDT to extract the knowledge carried inside the NK-DNA's *Experience.* During the training process, the proposed interpretable module transforms the implicit knowledge stored in the NK-DNA into explicitly represented decision trees via stochastic gradient descent, resulting in a transparent and explainable decision tree for knowledge sharing and reuse.

INITIAL EXPERIMENTS

A. Experiment Overview

We use the same maze problem used in the NK-DNA study (Zhang et al. 2017) to examine our proposed method. As shown in Figure 4, there are eight blocks in this maze. In the beginning, the agent knows nothing about the maze. It explores and learns the maze through four possible actions: *Up, Down, Left, and Right*. In each block, the agent can perform one of four possible operations. Finally, the agent should know the shortest way to *Block 8* in the maze.

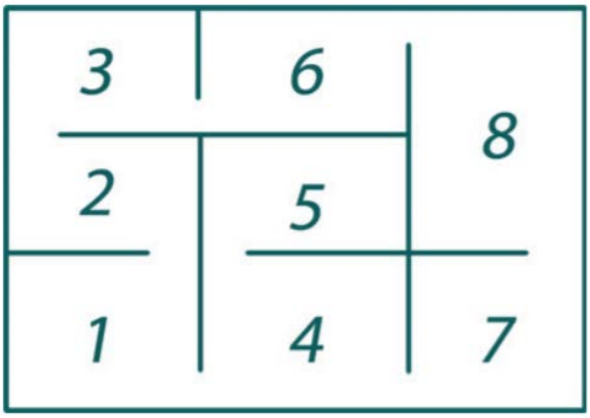

Fig. 4. The maze's environment

In our initial experiment, agents were asked to use the DRL algorithm to train the NK-DNA and the interpretable module to learn, store and reuse maze knowledge to find the shortest way.

B. Exploration

In this experiment, we set the agent always at *Block 1*. It randomly takes one available action from the four actions introduced above. Every step, the agent takes an action and gets feedback from the maze environment. Feedback consists of *Reward*, *Terminal*, and *Next State*: *Reward* is the value given by the maze after each action. If the agent reaches *Block 8*, the value will be 100 and otherwise is -1. The *Terminal* is a boolean value, if the agent goes *Block 8*, the value will be True, and the game is over. Otherwise, it is False, and the game continues. The *Next State* represents the block where the agent is after taking action.

The agent randomly repeats its possible operations until it reaches *Block 8*. At the same time, the agent stores every action with maze feedback as an experience of exploring the maze. The experience is stored in the form of (s_t, a_t, r_t, s_{t+1}): s_t represents the block where the agent is in at time step t; a_t is the action taken at that time step; r_t is the reward for taking that action, and s_{t+1} is the next state of the agent after performing the action a_t.

C. Training

In the exploration stage, the agent stores every action with feedback as an experience. While at the training stage, the agent uses the experience data to optimize the agent's policy for taking action (Equation 3) and use the gradient update algorithm to update the parameters of the DDTs for optimizing the policy (Equation 4).

$$f_T(s,a) = \mu(s)y_a^{True} + (1-\mu(s)y_a^{False}) \tag{3}$$

$$f_{T(s,a)} \rightarrow \pi(s,a) = \mu(s)\pi_a^{True} + (1-\mu(s))\pi_a^{False} \tag{4}$$

Then, the agent uses the optimal policy to generate a new experience from its interaction with the maze environment. Repeat that process until the policy guide the agent to find the best way from *Block 1* to *Block 8*, and we store the parameters of DDT and its detail of structure in a model. The training results are shown in Figure 5.

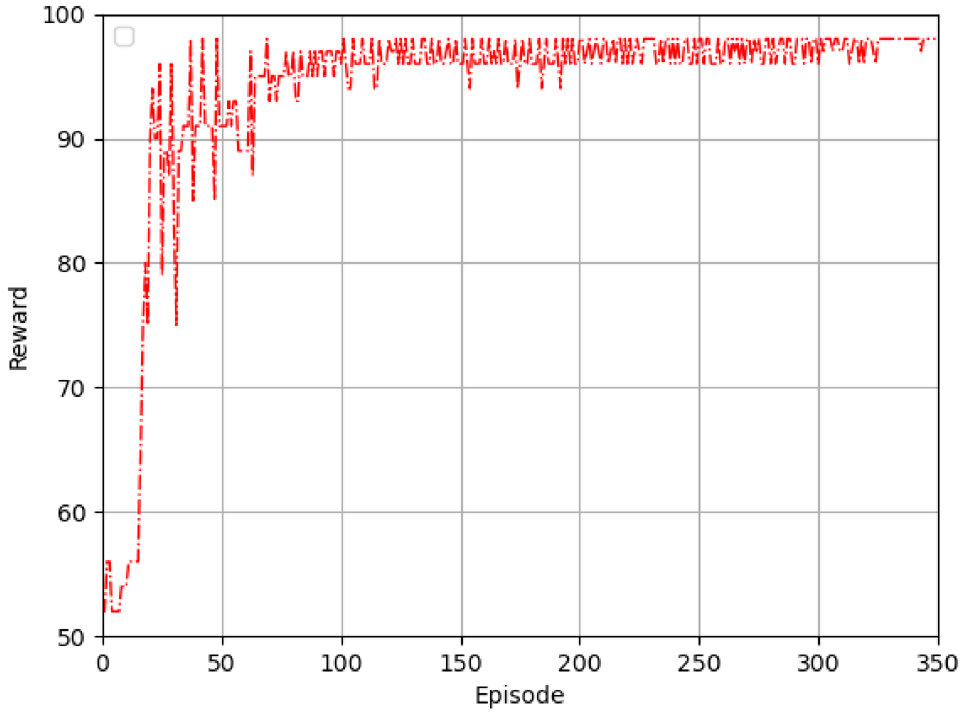

Fig. 5. The reward plots during training.

D. *Tree-structure Knowledge Extraction*

After training, we can choose the best model to generate a discrete decision tree to make the knowledge stored in the model easy to understand. Due to the property of the sigmoid function, it is not enough to ensure discrete decisions on each node. Therefore, in order to obtain a discrete tree, it is necessary to convert the differentiable tree to a discrete tree using the argmax function. We set the β_η to a one-hot vector for each non-leaf node, and divide φ to calculate and choose the max β_η for a specific non-leaf node. The φ is a bias value for comparison against the raw input feature x_i. Repeat the process for each node, and it finally creates a decision tree that compares a single feature per node and makes a single decision per leaf (see Figure 6).

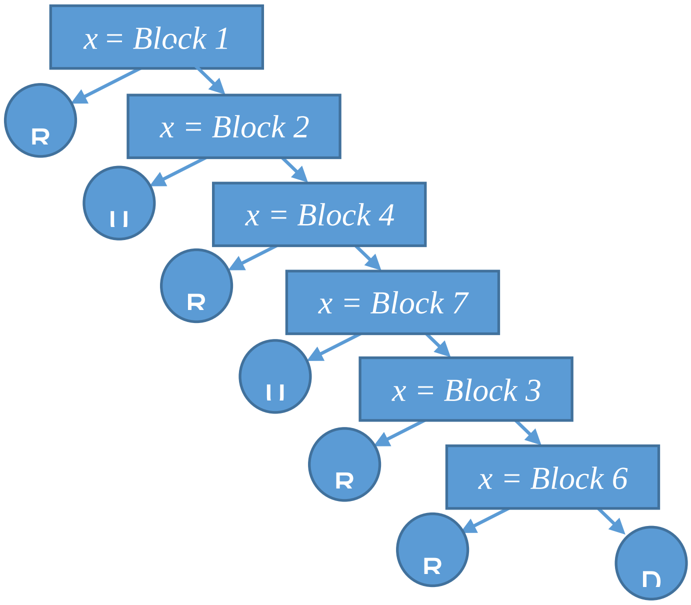

Fig. 6. The figure describes the generated decision tree. The tree goes to the left branch when the condition is True at the non-leaf node. At leaf nodes, there are actions: U means up, D means down, L means left, R means right. The *x* in the decision node is the agent's current state (position of the agent).

CONCLUSIONS AND FUTURE WORK

In this paper, we add an interpretable module to the NK-DNA to enhance its applicability and robustness. The proposed approach uses the modified decision tree and gradient descent to update the tree online with deep reinforcement learning. The experiment results show that the proposed method can transform the implicit knowledge stored in the NK-DNA into explicitly represented decision trees bringing fair interpretability to neural network-based intelligent systems.

For further work, we will refine this approach and apply it to complex tasks.

ACKNOWLEDGEMENT

The authors would like to thank the editors and anonymous reviewers for their valuable comments and suggestions on this paper. This work was supported by the Sichuan Science and Technology Program under Grant 2019YFH0185.

REFERENCES

Alberto Suárez and James F Lutsko. Globally optimal fuzzy decision trees for classification and regression. IEEE Transactions on Pattern Analysis and Machine Intelligence, 21(12):1297–1311, 1999.

Andrew Silva, Matthew Gombolay, Taylor Killian, Ivan Jimenez, Sung-Hyun Son. Optimization Methods for Interpretable Differentiable Decision Trees Applied to Reinforcement Learning. Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics, PMLR 108:1855-1865, 2020.

C. J. Kelly, A. Karthikesalingam, M. Suleyman, G. Corrado, and D. King, "Key challenges for delivering clinical impact with artificial intelligence," BMC Med., vol. 17, no. 1, p. 195, Dec. 2019.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. 2015. "Deep learning." Nature 521.7553: 436-444. DOI:10.1038/nature14539.

M. Puterman, Markov decision processes: Discrete stochastic dynamic programming, Technometrics 37(3) (1994), 353.

Podgorelec V, Kokol P, Stiglic B, Rozman I. Decision trees: an overview and their use in medicine. J Med Syst. 2002 Oct;26(5):445-63. DOI: 10.1023/a:1016409317640. PMID: 12182209.

R. Bellman, Dynamic Programming, Princeton University Press, 1957.

R.S. Sutton and A.G. Barto, Reinforcement Learning: An Introduction, MIT Press, 1998.

Sanin, C., and E. Szczerbicki. 2006. Using set of experience in the process of transforming information into knowledge. International Journal of Enterprise Information Systems 2 (2):40–55. doi:10.4018/jeis.2006040104.

S. Tonekaboni, S. Joshi, M. D. McCradden, and A. Goldenberg, "What clinicians want: Contextualizing explainable machine learning for clinical end use," CoRR, vol. abs/1905.05134, pp. 1–12, May 2019.

Tjoa E, Guan C. A Survey on Explainable Artificial Intelligence (XAI): Toward Medical XAI[J]. IEEE Transactions on Neural Networks and Learning Systems, 2020, PP.

Zhang, H., C. Sanin, E. Szczerbicki, and M. Zhu. 2017. Towards neural knowledge DNA. Journal of Intelligent & Fuzzy Systems 32 (2):1575–584. DOI:10.3233/jifs-169151.