Software Engineering Group
Department of Computer Science
Nanjing University
http://seg.nju.edu.cn

Technical Report No. NJU-SEG-2020-IC-006

2020-IC-006

Synthesizing ReLU Neural Networks with Two Hidden Layers as Barrier Certificates for Hybrid Systems

Qingye Zhao, Xin Chen, Yifan Zhang, Meng Sha, ZhengFeng Yang, Wang Lin, Enyi Tang,

Qiguang Chen, Xuandong Li

Technical Report 2020

prohibited.

Synthesizing ReLU Neural Networks with Two Hidden Layers as Barrier Certificates for Hybrid Systems

Qingye Zhao, Xin Chen*
Yifan Zhang, Meng Sha
State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China

Zhengfeng Yang
Shanghai Key Lab of Trustworthy Computing, East China Normal University, Shanghai, China

Wang Lin
School of Information Science and Technology, Zhejiang Sci-Tech University, Hangzhou, China

Enyi Tang
State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China

Qiguang Chen
British Columbia Academy, Nanjing Foreign Language School, Nanjing, China

Xuandong Li
State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China

ABSTRACT

Barrier certificates provide safety guarantees for hybrid systems. In this paper, we propose a novel approach to synthesize neural networks as barrier certificates. Candidate networks are trained from a special structure: ReLU neural networks consisting of two hidden layers. Then, the problem of identifying real barrier certificates from candidates is transformed into a group of mixed integer linear programming problems and a mixed integer quadratically constrained problem. Taking full advantage of the recent advance in optimization, barrier certificates validation can be performed effectively. We implement the tool *SyntheBC* and evaluate its performance over 3 hybrid systems and 8 continuous systems up to 12-dimensional state space. The experimental results show that our method is more scalable and effective than the classical polynomial barrier certificate method and the existing neural network based method.

CCS CONCEPTS

• **Computer systems organization** → **Embedded and cyber-physical systems**; • **General and reference** → **Verification**; • **Computing methodologies** → **Machine learning**.

KEYWORDS

hybrid systems, safety verification, barrier certificates, neural networks, mixed integer programming

ACM Reference Format:
Qingye Zhao, Xin Chen*, Yifan Zhang, Meng Sha, Zhengfeng Yang, Wang Lin, Enyi Tang, Qiguang Chen, and Xuandong Li. 2021. Synthesizing ReLU Neural Networks with Two Hidden Layers as Barrier Certificates for Hybrid Systems. In *24th ACM International Conference on Hybrid Systems: Computation and Control, May 19-21, 2021, Virtual.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

*Corresponding author:Xin Chen. chenxin@nju.edu.cn.

HSCC '21, May 19-21, 2021, Virtual

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Much research effort has been devoted to safety verification of non-linear hybrid systems as a result of the ever-growing safety requirement of complex embedded systems that consist of interacting computational and physical components [10, 27, 31, 40]. Safety verification contributes to checking safety properties by determining whether a system can evolve to some states violating the desired safety property when it starts at some initial states [22, 32, 34, 39]. A successful verification can raise our confidence in the verified system.

Safety properties can be directly ensured if it is proved that the exact reachable set or its over-approximation never invades the unsafe regions [1, 6, 33]. As the exact or approximate reachable set is derived step by step, these approaches are usually adopted in checking safety properties within a finite time horizon, and can hardly be used in verification concerning an infinite time horizon [1]. Besides, due to the intrinsic computational complexity, it is extremely difficult to scale them up to complex non-linear systems.

The methods resorting to barrier certificates are proposed to address the computational complexity and the infinite time horizon issues [13, 28]. A barrier certificate is a function of the state that separates reachable states of a system from its unsafe region. It requires all system trajectories starting from some initial states fall into one side of the barrier certificate while the unsafe region residents on the other. As the existence of a barrier certificate proves that the unsafe region is not reachable, the safety verification problem can be transformed into the problem of barrier certificate generation. Compared with reachable set computation, the construction of barrier certificates can easily treat the trajectories in an infinite time horizon and requires much less computational effort [20, 29, 48].

Barrier certificates of the type polynomial receive the most attention due to its ability to present complex non-linear curves [14, 21, 23, 29]. To find a polynomial barrier certificate, a polynomial with a fixed degree and unknown coefficients is introduced as a template. Then, all the verification conditions are encoded into a set of constraints on state variables and unknown coefficients of the

barrier certificate according to the theorem of positive semidefinite polynomials. Finally, those unknown coefficients are determined by solving the constraints.

To take advantage of the theorem of positive semidefinite polynomials, the systems under verification are not allowed to have non-polynomial terms. Systems with non-polynomial terms need to use their polynomial approximations and take care of the errors when they are verified. Furthermore, the polynomial with a high degree is very sensitive to errors and the number of unknown coefficients grows exponentially with respect to the degree of the template, which makes the generation of polynomials with a high degree not easy.

Recently, safety verification utilizing neural networks to separate the over-approximation of the reachable set from the unsafe region is proposed [26, 51]. Acting as barrier certificates, deep neural networks have the potential to define more complex barrier certificates than the classical polynomials do, as the theorem of universal approximation shows that a feed-forward neural network that comes with a single hidden layer comprising of a finite number of neurons can approximate any continues functions on compact subsets of R^n with enough precision [3, 12].

Similar to the classical polynomial barrier certificates that assign the over-approximation of the reachable region and the unsafe region with non-negative reals and negative reals, respectively, the neural network barrier certificates have the ability to distinguish the two regions with different signs.

The procedure for synthesizing neural network barrier certificates takes a data-driven approach. The candidate neural network barrier certificate is trained exploiting sampling data points coming from the initial state, the unsafe region, the system state space as well as some trajectories starting from the initial state. And then, it is verified according to the conditions that a real barrier certificate should satisfy. Thus, effective verification techniques are the key to the successful synthesis of barrier certificates.

Unfortunately, verification of neural network barrier certificates is not an easy task, as there may be a tremendous number of neurons and non-linear activation functions in them. Existing approaches resort to non-linear satisfiability modulo theories (SMT) solvers combined with the piece-wise linear approximation to identify real barrier certificates from candidate networks [51]. As all SMT solvers are not complete when handling non-linear arithmetic, plus the high computational complexity, those methods are prone to fail, which becomes an obstacle to neural network based verification.

In the paper, we propose the feedforward neural network of the special structure that consists of two hidden layers and uses the ReLU activation functions as barrier certificates. Such a special structure enables us to transform the problems of verifying barrier certificate conditions into mixed integer linear programming (MILP) and mixed integer quadratically constrained problems (MIQCP) with non-linear terms. Taking full advantage of the recent advance of the optimizer Gurobi 9.0 [11], which explores the entire search space and can provide an optimal objective value of either MILP or MIQCP problems, accompanied with its difference from the global optimal one, verification of the special network can be performed effectively and efficiently.

Our paper makes the following contributions:

- We adopt a special structure of feedforward neural network with two hidden layers and the ReLU activation functions to synthesize barrier certificates for safety verification of hybrid systems, and transforms the problem of certifying its conformance to barrier certificate conditions into a group of MILP and MIQCP problems.
- We develop an optimization based technique that takes advantages of the optimal objective value and its difference from the global optimal one, returned by the optimizer, to identify real barrier certificates from candidate networks.
- We implement the tool *SyntheBC* and evaluate its performance over a set of benchmark examples, which shows that our method can handle verification problems beyond the reach of classical polynomial barrier certificates, and is more scalable and effective than the existing neural network based verification method.

The rest of this paper is organized as follows. Section 2 gives an overview of related work. In Section 3, we briefly introduce some notions about hybrid systems and feedforward neural networks. Section 4 is devoted to discussing how to train a candidate neural network barrier certificate. Then we present the MIP-based encoding method for the safety verification conditions in Section 5. Experimental evaluation and comparisons with the classical sum-of-squares (SOS) method and the stat-of-the-art neural network based method are shown in Section 6. Section 7 concludes the paper.

2 RELATED WORK

Barrier certificate generation. Safety verification based on barrier certificates was first proposed by Prajna et al. in [28, 29], where the theory of Putinar's Positivstellensatz is exploited to form an SOS program of the barrier certificate. Kapinski et al. [13, 14] introduced a Lyapunov function typed barrier certificate. For semi-algebraic hybrid systems, Kong et al. [20, 21] proposed an exponential condition to generate barrier certificates. Sloth et al. [37] proposed a new barrier certificate for a special class of hybrid systems that consists of several interconnected subsystems. Dai et al. [4] studied how to relax the condition of barrier certificates in a general way while retaining the convexity. Tuncali et al. [43] combined a simulation-guided technique with a validation process to identify barrier certificates. Zeng et al. [50] presented the Darboux-type barrier certificates that define an algebraic curve, not allowing any trajectories of the system leaving it once they touch it. For hybrid systems containing elementary functions, Liu et al. [23] proposed a symbolic abstraction approach that replaces all non-polynomial terms with newly introduced variables so that they can be verified using the well-established verification techniques of polynomial hybrid systems. All the barrier certificates mentioned above are of the form polynomials.

To our best knowledge, the seminal work introducing neural networks as barrier certificates was proposed by Zhao et al. [51]. They exploit neural networks with Bent-ReLU activation functions to verify dynamical systems. After obtaining candidate networks, they first use piece-wise lines to over-approximate the Bent-ReLU function, and then invoke the non-linear SMT solver isat3 [45] to check the barrier certificate conditions. To get an over-approximation of Bent-ReLU function precise enough to enable

correct SMT judgement, the intervals of piece-wise lines should be carefully determined. They do not give an automatic interval determination method in the paper. In fact, to avoid the accumulation of approximation errors, in their experiments, all networks are of only one hidden layer where the number of neurons ranges from 5 to 20.

Peruffo et al. [26] presented a CEGIS-based technique that used counterexamples to speed up the construction of networks, serving as the barrier certificates for safety verification of hybrid systems. They used Tanh and polynomial activation functions and directly resorted to the non-linear SMT solver dReal and Z3 to check the barrier certificate conditions. Experiments showed their method was faster and required fewer data points in barrier certificate synthesis than Zhao's method.

Different from them, we adopt a special feedforward neural network structure that consists of two hidden layers and uses the ReLU activation functions to synthesize candidate barrier certificates, and employ a MIP optimization based technique to identify the real ones.

Neural network verification. The problem of neural network verification is NP-hard [15] and lots of research focuses on it [35, 41, 44, 46]. Existing methods for verifying neural network are based on abstract domain [36], abstract interpretation [35], input refinement [15], interval arithmetic [44], linear approximations [46] and mixed integer programming [5, 41, 42]. The previous MIP-based verification method is used to verify the output of neural network neurons on a given input region, which is a problem about mixed integer linear programming optimization. In this work, we adopt a MIP-based encode to check barrier certificate conditions. For those conditions determining neural network output, we encode them as several MILP optimization problems. Moreover, for the condition concerning the derivative of the network, it is encoded as a MIQCP optimization problem.

3 PRELIMINARIES

This section introduces the definitions used in the paper. Section 3.1 defines the continuous systems with safety verification conditions and trajectory. Section 3.2 defines the safety verification of the hybrid systems. Section 3.3 describes the feedforward neural networks.

3.1 Continuous system

Definition 1 (Continuous system). *A continuous system* Π *is a tuple* $\langle X, f, D, I\rangle$*, where* X *is a set of the system variables,* f *is the update functions over system variables* X*,* D *is the state space of* X*, and* $I \subseteq D$ *is the initial states.*

Assume the system update functions f satisfies the local Lipschitz condition, the trajectory of system Π is defined as follows:

Definition 2 (Trajectory). *Given a system* Π*, let* $\psi : [0, T] \times D \to D, T > 0$ *be the flow map satisfying:*

$$\psi(0, x) = x, x \in I \text{ and } \forall t \in [0, T], \frac{d\psi(t, x)}{dt} = f(\psi(t, x)).$$

Then, $\psi(t, x)$ *is the trajectory originating from an initial state* $x \in I$*. For every initial state* $x \in I$ *at time* t*,* $\psi(t, x) \in D$ *is unique.*

Given a continuous system Π and the unsafe region $U \subseteq D$, we want to verify whether the trajectories can reach U starting from some initial states. The following definition describes the safety of the continuous system Π.

Definition 3 (Safety). *Consider a continuous dynamical system* Π *with respect to the unsafe region* U*. The system* Π *is safe if* $\forall x \in I, \forall t \geq 0, \psi(t, x) \notin U$*, that is, all trajectories of the system starting from initial states* I *never reaches the unsafe region* U*.*

Barrier certificates can be used to verify safety properties without computing the set of reachable states explicitly. The essential idea is to use the zero level set of a barrier certificate $B(x)$ as a barrier to separate all the reachable states from the unsafe region. The safety verification conditions of barrier certificates are formulated as follows [28]:

Theorem 1. *A real barrier certificate* B *of the continuous system* Π *satisfies:*

$$B(x) > 0, \quad \forall x \in U, \tag{1}$$

$$B(x) \leq 0, \quad \forall x \in I, \tag{2}$$

$$\frac{\partial B(x)}{\partial x} f(x) \leq 0, \quad \forall B(x) = 0, \tag{3}$$

In Theorem 1, Equation (1) and (2) require B is positive and non-positive on the unsafe region and initial states, respectively. Equation (3) implies during continuous flow B can not jump to a positive state from a non-positive state. If Equation (1)-(3) are satisfied, B is a real barrier certificate of Π, and Π is safe.

3.2 Hybrid system

Definition 4 (Hybrid system). *A hybrid system* Π *is a tuple* $\langle M, X, f, D, I, T, R\rangle$*, where* $M : \{m_1, ..., m_k\}$ *is a finite set of modes,* X *is a set of the system variables,* f *is a set of update functions over system variables* X*,* D *is the state space,* $I \subseteq D$ *is a set of initial states,* $T : \{t_{m,m'} | m \neq m'\}$ *is a set of transition states between two modes, and* $R : \{r_{m,m'} | m \neq m'\}$ *is a set of reset functions corresponding to the terms in* T*.*

The trajectory of a hybrid system Π is similar to that of the continuous system shown in Definition 2. Given a hybrid system Π, we want to verify whether its trajectories starting from some initial states can reach the unsafe regions $U \subseteq D$. The conditions of barrier certificates for safety verification of hybrid systems are formulated as follows [28]:

Theorem 2. *For each* $m \in M$ *and* (m, m') *corresponding to the* $t_{m,m'}$ *and* $r_{m,m'}$*, a barrier certificate* B *of the hybrid system* Π *with respect to the unsafe regions* U *satisfies:*

$$B_m(x) > 0, \quad \forall x \in U_m, \tag{4}$$

$$B_m(x) \leq 0, \quad \forall x \in I_m, \tag{5}$$

$$\frac{\partial B_m(x)}{\partial x} f_m(x) \leq 0, \quad \forall B_m(x) = 0, \tag{6}$$

$$B_{m'}(x') \leq 0, \quad \forall x \in T_{m,m'}, B_m(x) \leq 0, x' = r_{m,m'}(x). \tag{7}$$

Theorem 2 requires that for each mode m, B_m separates unsafe regions and initial states with different signs, respectively. Equation (6) ensures that following the continuous flow, B_m is not

allowed to jump from a non-positive state to a positive one. Equation (7) guarantees B_m can not become positive after performing a discrete transition. If Equation (4)-(7) are satisfied for all the modes, B is a real barrier certificate of Π, where the safety of Π is certified.

In this paper, we focus on hybrid systems whose update functions are represented by multivariate elementary functions and variable states are semi-algebraic. Concretely, multivariate elementary functions are expressed by the following grammar:

$$f, g ::= c \mid x \mid f + g \mid f - g \mid f \times g \mid \frac{f}{g} \mid f^a \mid e^f \mid \ln(f) \mid \sin(f) \mid \cos(f),$$

where $c \in \mathbb{R}$ is a real constant, $a \in \mathbb{Q}$ is a rational constant, and $x \in X$ can be any system variable.

3.3 Feedforward neural network

In this work, we adopt feedforward neural networks as the representations of barrier certificates. The feedforward neural network consists of an input layer, an output layer, and multiple hidden layers in between. Neurons (so-called nodes) in the feedforward neural network are arranged in disjoint layers, with each neuron in one layer connected to the next layer, but no connection between neurons in the same layer. Furthermore, the output of each neuron in the hidden layer is assigned by a linear combination of the neuron outputs of the preceding layer and then applying a non-linear activation function. Formally, the feedforward neural network is defined as follows.

Definition 5 (Feedforward Neural Network). *A feedforward neural network $\mathcal{N}$ is a tuple $\langle L, W, B, \Phi, X \rangle$, where*

- *$L = \{L_0 \ldots, L_n\}$ is a set of layers, where layer L_0 is the input layer, L_n is the output layer, and $L_1, \ldots, L_{n-1}$ are the hidden layers. Each layer $L_k, 0 \leq k \leq n$ is associated with an s_k-dimensional vector space $\Psi_k \subseteq \mathbb{R}^{s_k}$, in which each dimension corresponds to a neuron.*
- *$W = \{W_1, \ldots, W_n\}$ is the set of weight matrices. Each non-input layer L_k with $1 \leq k \leq n$ has a weight matrix $W_k \in \mathbb{R}^{s_k \times s_{k-1}}$, and neurons in L_k are connected to neurons from the preceding layer L_{k-1} by the weight matrix W_k.*
- *$B = \{b_1, \ldots, b_n\}$ is the set of bias vectors. For each non-input layer L_k with $1 \leq k \leq n$, the bias vector $b_k \in \mathbb{R}^{s_k}$ is used to assigned bias values to the neurons in L_k.*
- *$\Phi = \{\phi_1, \ldots, \phi_n\}$ is a set of activation functions for hidden layers and are applied element-wise on each neuron.*
- *$X = \{x_0, \ldots, x_n\}$, where x_k is the vector corresponding to the values of the neurons in the layer L_k for $0 \leq k \leq n$. For each hidden layer, let z_k denote the neuron value vector before applying activation functions.*

For the above feedforward neural network, a non-input layer neuron value is computed by the preceding layer neuron values, the layer weight matrix, and the bias vector. Let x_0 denote the given neuron value of input layer; z_k and $x_k, 1 \leq k \leq n-1$ denote the neuron value of hidden layer L_k before and after activation function Φ_k, respectively; and x_n denote the neuron value of output layer,

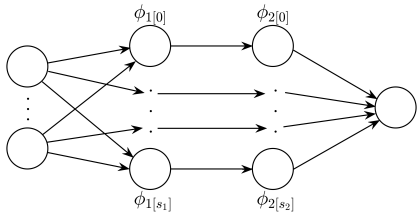

Figure 1: The structure of neural network for synthesizing barrier certificates. The neurons in input and output layers correspond to the system variables and the output of the barrier certificate. The two hidden layers use ReLU function as activation functions.

the forward propagation of the neural network is as follows:

$$\begin{cases} z_k = W_k x_{k-1} + b_k, & k = 1, ..., n-1, \\ x_k = \phi_k(z_k), & k = 1, ..., n-1, \\ x_n = W_n x_{n-1} + b_n. \end{cases} \tag{8}$$

The network output $\mathcal{N}(x)$ is the neuron value of the output layer, i.e., $\mathcal{N}(x_0) = x_n$ and x_n is computed as Equation (8).

4 BARRIER CERTIFICATE TRAINING

Given a hybrid system Π, our goal is to synthesize a neural network that satisfies the conditions in Theorem 2, so that it acts as a barrier certificate B, ensuring the unsafe region never to be invaded. Here, the process of synthesis consists of two steps. At first, a candidate neural network barrier certificate $\mathcal{N}$ is derived by training parameters of a special network structure, which uses the sampled data points from the initial state, the unsafe region, the system state space, and the trajectories as well. After that, the network $\mathcal{N}$ is verified according to the barrier certificate conditions defined in Theorem 2, where the special structure of the network enables the transformation of verification problem into a set of MILP and MIQCP problems. In the section, we detail the training process while the verification technique is presented in the next section.

4.1 A Special Structure of Neural Networks

We propose a feedforward neural network of a special structure as the template for synthesizing barrier certificates, which is characterized as follows:

- The input layer contains the same number of neurons as that of the system variables, and each system variable corresponds to one unique neuron of the input layer;
- The output layer has only one neuron, producing the output of the barrier certificate.
- There are two and only two hidden layers, each of which can have many neurons.
- The ReLU function is the only legal activation function.

The ReLU functions is defined as follows:

$$\text{ReLU}(x) = \begin{cases} x, & \text{if } x > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

The special structure of networks is depicted in Figure 1. The design of the special network structure tries to balance the requirement of enough expressive power to approximate complex functions and easy verification.

By restricting the number of hidden layers and the type of activation function, the problem of verifying Equation (4), Equation (5), and Equation (7) in Theorem 2 forms a group of MILP problems, while the problem of certifying Equation (6) forms a MIQCP problem.

Although, theorem of universal approximation [3, 12] has proven that one hidden layer with a large number of neurons is good enough to obtain precise approximation to complex functions, recent research on neural network training has shown that a network with deep architectures provides better expressive power and are easier to be trained [24, 30]. Here, we set the number of hidden layers to 2 to obtain the global the optimal objective value of the problems of the two types.

4.2 Training dataset generation

The procedure for synthesizing neural networks takes a data-driven approach. Here, we discuss how to generate adequate datasets for training the candidate network. Datasets are generated aiming at making the network fulfill the barrier certificate conditions given by Theorem 2.

The first dataset is designed to let the network output desired results on different regions, corresponding to Equation (4), Equation (5). Let dataset $D_1 : \{x \in U_m\}$ contains data points sampled from unsafe regions, according to Equation (4), when fed with points in D_1, the network should output positive values. Similarly, let $D_2 : \{x \in I_m\}$ consists of data points sampled from initial states, the network should output non-positive values for these points.

Equation (7) claims that discrete jumps should retain non-positive signs. To make the network conform to this condition, we have to exploit system trajectories. By choosing a time bound T, which lasts long enough, some trajectories starting from initial state trigger several discrete jumps. We sample date points from those trajectories $\psi(t, x), t \in [0, T]$, and add them to the dataset D_2, as all of them should make the network produce non-positive values.

The last dataset D_3 is defined as $\{x|\mathcal{N}(x) = 0 \wedge x \in D\}$, which is used to train the network to satisfy the condition $\frac{\partial \mathcal{N}(x)}{\partial x} f(x) \leq 0$, when $\mathcal{N}(x) = 0$. Note that, the network parameters keep on updating during the training, so the set keeps on change its elements accordingly. Given a network, it requires unaffordable computational effort to build dataset D_3. Thus we use a testing based approach, which randomly samples points from the system state space D, and add those that make the network evaluate to zero to D_3 .

4.3 Loss function construction

Utilizing the datasets D_1, D_2 and D_3, the neural network is trained to satisfy the following conditions:

$$\mathcal{N}(x) > 0, \quad \forall x \in D_1, \tag{10}$$

$$\mathcal{N}(x) \leq 0, \quad \forall x \in D_2, \tag{11}$$

$$\frac{\partial \mathcal{N}(x)}{\partial x} f(x) \leq 0, \quad \forall x \in D_3. \tag{12}$$

According to Equation (10)-(12), we construct sub loss functions that lead optimizers to build the network by minimizing the loss.

To meet Equation (10), the neural network $\mathcal{N}(x)$ should output positive values when the input comes from $D1$. Otherwise, it will produce loss. Let $\epsilon_1 > 0$ be a small positive number, the first sub loss function is defined as follows:

$$l_1 = \sum_{x \in D_1} -\min(\mathcal{N}(x) - \epsilon_1, 0). \tag{13}$$

Similarly, to conform to Equation (11), the neural network $\mathcal{N}(x)$ must output negative values. Otherwise, it will produce loss too. The second sub loss function is defined as follows:

$$l_2 = \sum_{x \in D_2} \max(\mathcal{N}(x), 0). \tag{14}$$

For Equation (12), the constraint $\mathcal{N}(x) = 0$ should be relaxed first, due to the errors introduced by real number computation. Let $\epsilon_2 > 0$ be a small positive number denoting the threshold of relaxation, the constraint $\mathcal{N}(x) = 0$ is relaxed as $-\epsilon_2 \leq \mathcal{N}(x) \leq 0$. As a result, the dataset D_3 is redefined as $\{x| - \epsilon_2 \leq \mathcal{N}(x) \leq 0 \wedge x \in D\}$. The third sub loss function is defined as follows:

$$l_3 = \begin{cases} \sum \max(\frac{\partial \mathcal{N}(x)}{\partial x} f(x), 0), & x \in D_3, \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

Finally, let $\alpha > 0, \beta > 0$, and $\gamma > 0$ denote the weights of sub losses, respectively, the overall loss function is defined as the weighted sum of sub losses:

$$l = \alpha l_1 + \beta l_2 + \gamma l_3. \tag{16}$$

We minimize l to train a candidate neural network barrier certificate using classical Adam optimizers [17].

5 BARRIER CERTIFICATES VERIFICATION

In this section, we focus on how to identify real barrier certificates from the candidate neural networks yielded from the last section. The key point is to utilize the special structure of the network to encode the verification problem as several optimization problems, and then resort to the optimizer to find the global optimal objective value.

Suppose a network $\mathcal{N}$ is of the special structure introduced in Section 4.1, to ease the presentation, we use x_0 to denote the input of the network and denote the neural network output as x_3, where $x_3 = \mathcal{N}(x_0)$, since the third layer of $\mathcal{N}$ is the output layer and it consists of only one neuron.

To become a barrier certificate, the network $\mathcal{N}$ must satisfy the following conditions given by Theorem 3.

Theorem 3. *Let the neural network $\mathcal{N}$ accept the input x_0, which corresponds to the system variables of X of the hybrid system Π. Given the unsafe regions U, $\mathcal{N}$ acts as a barrier function of Π, if it satisfies the following conditions:*

$$x_3 > 0, \quad \forall x_0 \in U_m, \tag{17}$$

$$x_3 \leq 0, \quad \forall x_0 \in I_m, \tag{18}$$

$$\frac{\partial x_3}{\partial x_0} f_m(x_0) \leq 0, \quad \forall x_3 = 0, \tag{19}$$

$$x_3' \leq 0, \quad \forall x_0 \in t_{m,m'}, x_3 \leq 0, x_0' = r_{m,m'}(x_0). \tag{20}$$

The barrier certificate conditions in Theorem 3 forms verification problems of two types: Equation (17), (18) and (20) require to check the output range of the network with respect to the given input region while Equation (19) cares for the derivative of the network

on the zero level set of the network. In the following, we encode them to a group of MILP optimization problems and a MIQCP optimization problem, respectively.

5.1 Verifying neural network output using MILP-based encoding

We begin with the verification of the condition: the network should produce positive values for the unsafe region, formulated in Equation (17). By adopting the forward propagation of neural networks described in Section 3, the problem of verifying the Equation (17) is transformed into the following optimization problem:

$$\begin{array}{lll} p = \min x_3 & & \\ \text{s.t.} \quad x_0 \in U_m, & \forall m \in M, & \\ \quad z_k = W_k x_{k-1} + b_k, & k = 1, 2, & \\ \quad x_k = \text{ReLU}(z_k), & k = 1, 2, & \\ \quad x_3 = W_3 x_2 + b_3. & & \end{array} \tag{21}$$

Suppose p^* is the global optimal solution of the problem (21), if $p^* > 0$, Equation (17) is verified to be satisfied.

It is intractable to directly optimize the problem (21) due to the non-linearity coming from the activation function ReLU and the constraints defining U_m. An exact MILP-based encoding of ReLU is introduced to eliminate the non-linearity that activation functions bring.

MILP-based encoding of ReLU. The MILP-based encoding presented in the following theorem has been widely used to treat ReLU [42].

Theorem 4. *Let $[l_x, u_x]$ denote the region of x, the ReLU function can be encoded by the following linear constraints:*

$$-x + y - l \cdot t \leq -l, \tag{22}$$

$$x - y \leq 0, \tag{23}$$

$$y - u \cdot t \leq 0, \tag{24}$$

with linear constraint and binary constraint:

$$y \geq 0, \tag{25}$$

$$t \in \{0, 1\}, \tag{26}$$

where t is the intermediate binary variable, l and u are bound estimation constants satisfying $l \leq l_x$ and $u \geq u_x$, respectively.

Piece-wise linear approximation. To handle the non-linearity arising from the constraints defining the unsafe region U_m, a piece-wise linear approximation method is utilized. For a non-linear function $g(x)$ with $x \in [l_x, u_x]$, it can be approximated by two piece-wise linear functions $g_l(x)$ and $g_u(x)$, where the difference between $g_l(x)$ and $g_u(x)$ is not greater than the controlled approximation error η, formally defined as:

$$g_l(x) \leq g(x) \leq g_u(x), |g_u(x) - g_l(x)| \leq \eta, \forall x \in [l_x, u_x]. \tag{27}$$

MILP-based encoding of Equation (17). Using the piece-wise linear approximation method, U can be approximated by two sets of piece-wise linear functions g_l and g_u, i.e., $g_{l[m]} \leq U_m \leq g_{u[m]}$. Together with the MILP-based encoding of ReLU, the optimization problem (21) is further transformed into the following MILP problem:

$$\begin{array}{lll} p = \min x_3 & & \\ \text{s.t.} \quad g_{l[m]} \leq x_0 \leq g_{u[m]}, & \forall m \in M, & \\ \quad z_k = W_k x_{k-1} + b_k, & k = 1, 2, & \\ \quad -z_k + x_k - l_k t_k \leq -l_k, & k = 1, 2, & \\ \quad z_k - x_k \leq 0, & k = 1, 2, & \\ \quad x_k - u_k t_k \leq 0, & k = 1, 2, & \\ \quad x_k \geq 0, & k = 1, 2, & \\ \quad t_k \in \{0, 1\}^{s_k}, & k = 1, 2, & \\ \quad x_3 = W_3 x_2 + b_3, & & \end{array} \tag{28}$$

where the bound constants l_k and u_k can be estimated using interval computation [16] or simply set to the numbers small enough ($l_k = -1e9$) and large enough ($u_k = 1e9$), respectively.

Remark 1. *As a result of the piece-wise linear approximation applied to U_m, compared with the problem (21), the feasible set of problem (28) is the superset of that of (21), which indicates that the optimal solution p_r^* of (28) is the lower bound of the optimal solution p^* of (21), i.e., $p_r^* \leq p^*$.*

In practice, given a region defined by a set of nonlinear constraints, it is not easy to construct its piece-wise linear approximation. Fortunately, the optimizer Gurobi provides automatical piece-wise linear approximation whose error under the control of the parameter η in its recent version 9.0. So we do not need to consider the approximation of U_m in hand anymore.

Getting guaranteed solution. The error introduced by piece-wise linear approximation and the error accompanied with float point number calculation make it impossible for the optimizer to find the real global optimum, although given enough time, the global optimum of MILP problems can be found by searching the whole space.

In fact, for the problem (28), Gurobi can return an optimum p', together with the maximum relative error ξ between p' and the global optimal solution p^*, formally:

$$\xi \geq \frac{|p' - p^*|}{|p'|}. \tag{29}$$

This feature allows us to certify the original problem based on the an optimum p' and the accompanied maximum relative error ξ. The following theorem connects the return of Gurobi with the verification of Equation (17).

Theorem 5. *When solving problem (28), if the solution $p' > 0$ while the maximum relative error $\xi < 1$, then the global optimal solution p_r^* of problem (28) is positive, indicating that the Equation (17) is verified to be satisfied.*

The proof of Theorem 5 uses the following lemma.

Lemma 6. *Let p' be one optima of problem (28), p_r^* be the global optima, $\frac{|p'-p_r^*|}{|p'|} \leq \xi$, if $\xi < 1$, then p' and p^* have the same signs.*

Proof.

$$\begin{array}{lll} & \xi < 1 & \\ \Rightarrow & 1 > \frac{|p'-p_r^*|}{|p'|} & (\xi \geq \frac{|p'-p_r^*|}{|p'|}) \\ \Rightarrow & |p' - p_r^*|^2 < |p'|^2 & (|p' - p_r^*| > 0, |p'| > 0) \\ \Rightarrow & (p')^2 - 2p'p_r^* + (p_r^*)^2 < (p')^2 & \\ \Rightarrow & p'p_r^* > \frac{1}{2}(p_r^*)^2 \geq 0 & \end{array} \tag{30}$$

□

Theorem 5 can be proved as follows: according to lemma 6, p' and p_r^* have the same signs as $\xi < 1$. Due to the condition $p' > 0$, the global optimal solution $p_r^* > 0$. As Remark 1 states that $p^* \geq p_r^* > 0$, the global optimal solution p^* of problem (21) retains positive. Thus Equation (17) is verified to be satisfied.

MILP-based encoding of Equation (18). The verification of Equation (18) takes the same manner, as Equation (18) has the same structure as that of Equation (17) except for the input and output of the network. It is expected that the network should output non-positive values from the input of the initial set.

Exploiting two sets of piece-wise linear functions g_l and g_u to approximate the initial state I, i.e., $g_{l[m]} \leq I_m \leq g_{u[m]}$, combined with MILP encoding, the Equation (18) corresponds to the following optimization problem:

$$\begin{array}{lll} p = \max x_3 & & \\ \text{s.t.} \quad g_{l[m]} \leq x_0 \leq g_{u[m]}, & & \forall m \in M, \\ \quad z_k = W_k x_{k-1} + b_k, & & k = 1, 2, \\ \quad -z_k + x_k - l_k t_k \leq -l_k, & & k = 1, 2, \\ \quad z_k - x_k \leq 0, & & k = 1, 2, \\ \quad x_k - u_k t_k \leq 0, & & k = 1, 2, \\ \quad x_k \geq 0, & & k = 1, 2, \\ \quad t_k \in \{0,1\}^{s_k}, & & k = 1, 2, \\ \quad x_3 = W_3 x_2 + b_3. & & \end{array} \tag{31}$$

Verification of Equation (18) is based on the following theorem.

Theorem 7. *When solving problem (31), if the solution $p' \leq 0$ while the maximum relative error $\xi < 1$, then the global optimal solution p^* of problem (31) is non-positive, so that the Equation (18) is ensured to be satisfied.*

MILP-based encoding of Equation (20). For Equation (20), there may exist non-linear terms in transition states T and reset functions R, which should be approximated by piece-wise linear functions. Besides, the encoding of Equation (20) is similar to Equation (17), containing two parts. The first one is used to encode the constraints before reset functions, i.e., concerning x_0. The second one is used to encode the constraints after reset functions, i.e., concerning $x_0' = r_{m,m'}(x_0)$. For the transition state $t_{m,m'}$ and reset function $r_{m,m'}$, the encoding of Equation (20) is given as follows:

$$\begin{array}{llll} p = \max x_3' & & & \\ \text{s.t. } x_0 \in t_{m,m'}, & x_0' = r_{m,m'}(x_0) & & \forall t_{m,m'} \in T, \\ \quad z_k = W_k x_{k-1} + b_k, & z_k' = W_k x_{k-1}' + b_k, & & k = 1, 2, \\ \quad -z_k + x_k - l_k t_k \leq -l_k, & -z_k' + x_k' - l_k' t_k' \leq -l_k', & & k = 1, 2, \\ \quad z_k - x_k \leq 0, & z_k' - x_k' \leq 0, & & k = 1, 2, \\ \quad x_k - u_k t_k \leq 0, & x_k' - u_k' t_k' \leq 0, & & k = 1, 2, \\ \quad x_k \geq 0, & x_k' \geq 0, & & k = 1, 2, \\ \quad t_k \in \{0,1\}^{s_k}, & t_k' \in \{0,1\}^{s_k}, & & k = 1, 2, \\ \quad W_3 x_2 + b_3 \leq 0, & x_3' = W_3 x_2' + b_3, & & \end{array} \tag{32}$$

where the left side and the right side are the networks serving as barrier certificates on mode m and m', respectively.

Theorem 8 supports the verification of Equation (20).

Theorem 8. *When solving problem (32), if the solution $p' \leq 0$ while the maximum relative error $\xi < 1$, then the global optimal solution p^* of problem (32) is non-positive, so that the Equation (20) is ensured to be satisfied.*

5.2 Verifying the derivative condition using MIQCP-based encoding

Now we turn to the verification of Equation (19). For the constraint $\forall x_3 = 0$ in Equation (19), it can be encoded as $x_3 = 0$ and $x_0 \in D$, plus the encoding of the network used in problem (21). For the derivative $\frac{\partial x_3}{\partial x_0} f_m(x_0)$, the update function $f_m(x_0)$ is defined by system Π and is going to be approximated by piece-wise linear functions. In the following, we focus on the representation of the derivative of the network, i.e., $\frac{\partial x_3}{\partial x_0}$.

Theorem 9. *Let $r(x)$ is the derivative of ReLU(x), represented as:*

$$r(x) = \begin{cases} 1, & \text{if } x > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{33}$$

Let $x_{[i]}$ denote the i-th element of the vectors x, and $W_r(x) = \boldsymbol{diag}(r(x_{[1]}), r(x_{[2]}), ..., r(x_{[n]}))$ be a diagonal matrix, the derivative of the network with respect to the input x_0, denoted by $\frac{\partial x_3}{\partial x_0}$, can be derived as follows:

$$\frac{\partial x_3}{\partial x_0} = W_3 W_r(z_2) W_2 W_r(z_1) W_1, \tag{34}$$

where z_k denotes the value of neurons in hidden layers before applying the ReLU function.

Proof. Let $z_k = W_k x_{k-1} + b_k$ and $x_k = \text{ReLU}(z_k)$, $k = 1, 2$, denote the neuron value before and after applying ReLU, as shown in Equation (8), respectively. For different neuron network layers, we have the following backward propagation:

$$\begin{cases} \frac{\partial x_3}{\partial x_2} = W_3 \\ \frac{\partial x_k}{\partial z_k} = \frac{\partial \Phi_k(z_k)}{\partial z_k}, & k = 1, 2, \\ \frac{\partial z_k}{\partial x_{k-1}} = W_k, & k = 1, 2. \end{cases} \tag{35}$$

Referring to Equation (33): $\frac{\partial \Phi_k(z_k)}{\partial z_k} = r(z_k)$. Applying the chain rule, the derivative $\frac{\partial x_3}{\partial x_0}$ is yielded by:

$$\begin{aligned} \frac{\partial x_3}{\partial x_0} &= \frac{\partial x_3}{\partial x_2} \frac{\partial x_2}{\partial z_2} \frac{\partial z_2}{\partial x_1} \frac{\partial x_1}{\partial z_1} \frac{\partial z_1}{\partial x_0} \\ &= W_3 W_r(z_2) W_2 W_r(z_1) W_1. \end{aligned}$$

□

The following theorem further refines the derivation by taking advantage of the MILP-based encoding of the ReLU function.

Theorem 10. *Let ReLU function is represented by the MILP encoding in Theorem 4, $\frac{\partial x_3}{\partial x_0}$ can be further refined as:*

$$\frac{\partial x_3}{\partial x_0} = W_3 \cdot t_2 W_2 \cdot t_1 W_1, \tag{36}$$

where $\cdot$ denotes the element-wise product.

Proof. The MILP-based ReLU encoding in Theorem 4 shows the fact that $t = 0 \Leftrightarrow x \leq 0$ and $t = 1 \Leftrightarrow x > 0$. Combining with Equation (33): $r(x) = 0 \Leftrightarrow x \leq 0$ and $r(x) = 1 \Leftrightarrow x > 0$, it is easy to see that the value of binary variable t is exactly the derivative value r of ReLU, i.e., $t = r(x)$. Therefore, by replacing the diagonal matrix $W_r(x)$ with t in Equation (34), we get a representation of $\frac{\partial x_3}{\partial x_0}$ in terms of t. □

Equation (36) discloses the fact that $\frac{\partial x_3}{\partial x_0}$ is a function defined over the binary variable t, as W_k is the determined network parameters. Furthermore, each hidden layer L_k with ReLU activation functions contributes a vector of binary variables t_k, so the degree of $\frac{\partial x_3}{\partial x_0}$ is determined by the number of hidden layers.

In general, neural networks with more hidden layers are supposed to have stronger expressive power. However, the higher the degree is, the more difficult the verification is. We choose the ReLU neural network with two hidden layers as the template of candidate barrier certificates, as shown in Section 4.1. Under the structural configuration, the degree of $\frac{\partial x_3}{\partial x_0}$ is 2, where there are two binary vectors t_1 and t_2.

Now we proceed to $\frac{\partial x_3}{\partial x_0} f_m(x)$, the objective of the following optimization problem. Let $x_{0[i]}$ and $f_{m[i]}(x)$ denote the i-th element of the vectors x_0 and $f_m(x)$, respectively, the product of $\frac{\partial x_3}{\partial x_0}$ and $f_m(x)$ is the sum of the product of correspondence elements defined as:

$$\frac{\partial x_3}{\partial x_0} f_m(x) = \sum_{i=1}^{|X|} \frac{\partial x_3}{\partial x_{0[i]}} f_{m[i]}(x), \tag{37}$$

where $|X|$ is the number of system variables, also the number of neurons in the input layer.

An intermediate variable v is introduced as a constraint of the optimization problem, which is a quadratic constraint about two binary vectors t_1 and t_2, define as: $v = W_3 \cdot t_2 W_2 \cdot t_1 W_1$. Then, the optimization objective reacting to Equation (19) becomes $v f_m(x)$. Here, v is a variable of degree one and $f_m(x)$ can be over-approximated by piece-wise linear functions with degree one, so the optimization objective $v f_m(x)$ is quadratic. We construct the following MIQCP problem for verifying the derivative condition:

$$\left.\begin{array}{lll} p = \max\ v f_m(x) & & \\ \text{s.t.} \quad x_0 \in D_m, & & \forall m \in M, \\ \quad z_k = W_k x_{k-1} + b_k, & & k = 1, 2, \\ \quad -z_k + x_k - l_k t_k \le -l_k, & & k = 1, 2, \\ \quad z_k - x_k \le 0, & & k = 1, 2, \\ \quad x_k - u_k t_k \le 0, & & k = 1, 2, \\ \quad x_k \ge 0, & & k = 1, 2, \\ \quad t_k \in \{0, 1\}^{s_k}, & & k = 1, 2, \\ \quad W_3 x_2 + b_3 = 0, & & \\ \quad v = W_3 \cdot t_2 W_2 \cdot t_1 W_1, & & \end{array}\right\} \tag{38}$$

where l_k and u_k are estimated bound constants and $f_m(x)$ and D_m are approximated with the piece-wise linear functions.

For MIQCP problems, the optimizer Gurobi can search the whole state space, and return an optimum with its maximum relative error to the global optimum. Likely, we use the following theorem to certify the condition.

Theorem 11. *For the MIQCP problem (38), if the solution $p' \le 0$ while the maximum relative error $\xi < 1$, then the global optimal solution p^* of problem (38) is non-positive. For $\forall x_0 \in D_m$ such that $x_3 = 0$, we have $\frac{\partial x_3}{\partial x_0} f_m(x) \le 0$ and Equation (19) is verified to be satisfied.*

5.3 Algorithm

The detailed barrier certificate synthesis algorithm is presented in Algorithm 1. To train a neural network barrier certificate, we first generate the training datasets and initialize the neural network $\mathcal{N}$ (Line 1-2). In each epoch, the training datasets are divided into several batches and loss function l are constructed on each batch (Line 3-6). The network is trained by minimizing l using Adam optimizer (Line 7). Note that loss function minimization optimization is out of the scope of this paper, and we direct readers to [2, 8, 38]. After training on all batches, $\mathcal{N}$ is regarded as the candidate barrier certificate. The safety verification of the candidate neural network barrier certificate $\mathcal{N}$ is encoded to a set of MIP-based problems. The verification of requiring positive for unsafe regions, non-positive for initial states, and non-positive after discrete transitions is transformed into a group of MILP problems as according to (28), (31), and (32), respectively (Line 8). The verification of requiring non-positive about network derivative when following the continuous flow is transformed into the MIQCP problem according to (38) (Line 9). These MIP-based optimization problems are solved by the MIP solver to get the current solutions p' with the maximum relative error ξ (Line 10). If $\xi < 1$ and p' have the expected signs, i.e., positive sign for problem (28) and non-positive signs for problem (31), (32), (38), $\mathcal{N}$ is a real barrier certificate and return *success* and $\mathcal{N}$ (Line 12). Otherwise, sampling data points around counterexamples returned by optimizer and add them to the corresponding training datasets to enhance the training datasets (Line 14). The above barrier certificate synthesis process continues until finding a real barrier certificate or reaching the maximum epoch number.

Algorithm 1: Barrier Certificate Synthesis

Input: Hybrid system Π, trajectory time bound T, maximum epoch number e_{max}, loss function parameters $\epsilon_1, \epsilon_2, \alpha, \beta, \gamma$
Output: Result flag $flag$ and barrier certificate $\mathcal{N}$

1 Generate datasets D_1, D_2, D_3 with trajectory points in $[0, T]$
2 Initialize parameters θ of neural network $\mathcal{N}$
3 **for** $e = 1$ *to* e_{max} **do**
4 Divide D_1, D_2, D_3 to batches $d_s : \{d_1, ..., d_n\}$
5 **for** *each* $d_i \in d_s$ **do**
6 Construct loss function l with $\epsilon_1, \epsilon_2, \alpha, \beta, \gamma$ according to Equation (16)
7 Update θ by minimizing l using Adam optimizer
8 Transform network output verification into the MILP problems according to (28) (31) (32)
9 Transform network derivative verification into the MIQCP problem according to (38)
10 Optimize above MIP problems to get current solutions p' with the maximum relative error ξ
11 **if** p' *have the expected signs and* $\xi < 1$ **then**
12 Return *success*, $\mathcal{N}$
13 **else**
14 Sample data points around verification counterexamples and add them to training datasets
15 Return $failure, None$

6 EXPERIMENTS

We have implemented a barrier certificate synthesis tool named *SyntheBC* based on Algorithm 1. In this section, we present an experimental evaluation of *SyntheBC* over a set of benchmark examples, including 3 hybrid systems and 8 continuous systems up to 12-dimensional state space, and compare with the neural network barrier certificate synthesized framework *nnbarrier* [51] and the

classical SOS barrier certificate generation method [19]. For *nnbarrier*, we use their default settings. For the SOS method, we explore the different degrees of polynomial barrier certificate up to 8. We implement *SyntheBC* based on Tensorflow 1.14 and Gurobi 9.0. All experiments are conducted on a machine running Ubuntu 16.04 with 128GB RAM, a 3.20GHz Intel Xeon Gold 6146 CPU, and an NVIDIA TITAN V GPU.

In experiments, the examples contain elementary functions such as *pow*, *sqrt*, *exp*, *sin*, *cos* and the number of system variables reaches up to 12. For all examples, we set the parameters $\epsilon_1 = 0.0001, \epsilon_2 = 0.001, \alpha = \beta = \gamma = 1$ and $\delta = 0.5$.

6.1 Case studies

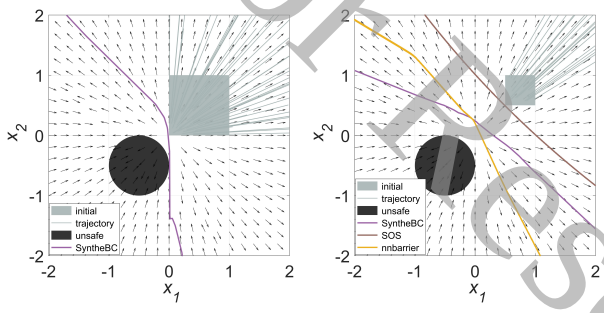

Figure 2: The original system and changed system in Example 1. Left subgraph plots the neural network barrier certificate synthesized by *SyntheBC* on the original system. Right subgraph plots barrier certificates synthesized by *SyntheBC*, the SOS method, and *nnbarrier* on the changed system with 1/4 initial state.

Example 1. *Consider the following continuous system [25]:*

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} x_1^2 + x_1x_2 + x_1 \\ x_1x_2 + x_2^2 + x_2 \end{bmatrix},$$

with the state space:

$$D = \{-2 \le x_1, x_2 \le 2\}.$$

It is required to verify that all trajectories of the system starting from the initial states:

$$I = \{0 \le x_1, x_2 \le 1\},$$

will never enter the unsafe region:

$$U = \{(x_1 + 0.5)^2 + (x_2 + 0.5)^2 \le 0.25\}.$$

A neural network $\mathcal{N}$ with ReLU activation functions of the structure 2-20-16-1 is trained as the barrier certificate by SyntheBC*, where the number of neurons of each layer is separated by '-'. Initially, each training data set is generated with 100,000 randomly sampled data points. SyntheBC randomly samples 300 points from initial states as the starting point of the trajectories and move forward 100 steps with step size 0.1.*

The left subgraph in Figure 2 displays the initial state, trajectories, the unsafe region of the system, and the zero level set of the neural network barrier certificate synthesized by SyntheBC*. The network curve is flexible enough to turn smoothly near the zero point and walk along with the line $x_1 = 0$. For* nnbarrier *and the SOS method, they fail to synthesize barrier certificates.*

For a more comprehensive comparison, we reduce the initial state to one quarter of the original one, all methods can synthesize barrier certificates as shown in the right subfigure in Figure 2. In this case, the barrier certificates synthesized by nnbarrier *and* SyntheBC *have similar behaviors, while the SOS method synthesizes a barrier certificate close to the initial state. This example shows that* SyntheBC *is more capable and works well in extreme cases.*

Example 2. *Consider the HIV transmission model [23]:*

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} -\frac{\beta c x_1 x_2}{x_1+x_2+x_3} - \mu x_1 \\ \frac{\beta c x_1 x_2}{x_1+x_2+x_3} - (\mu+\nu)x_2 \\ \nu x_2 - \alpha x_3 \end{bmatrix},$$

where x_1, x_2, and x_3 denote the part of the population: HIV susceptible, HIV infected, and AIDS diagnosed, respectively; β is the possibility of infection per partner contact; c is the rate of partner change; μ is the death rate of non-AIDS population; α is the death rate of AIDS patients; and ν is the rate at which HIV infected people develop AIDS. As shown in [23], the parameters are: $\beta = 0.2$, $c = 10$, $\mu = 0.008$, $\alpha = 0.95$, and $\nu = 0.1$. Suppose the system state space D is:

$$D = \{0 \le x_1, x_2, x_3 \le 10.013\}.$$

The goal is to verify that all trajectories of the system starting from the initial states:

$$I = \{9.985 \le x_1 \le 9.995, 0.005 \le x_2 \le 0.015, 0 \le x_3 \le 0.003\},$$

will never enter the unsafe region:

$$U = \{0 \le x_1, x_2 \le 10.013, 1 \le x_3 \le 10.013\}.$$

For this example, SyntheBC *and* nnbarrier *fails to synthesize barrier certificates and only the SOS method succeeds. The possible reason for* SyntheBC*'s failure is the asymmetric size of the initial states and the insecure region.*

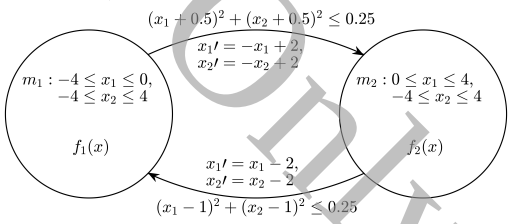

Figure 3: The hybrid system in Example 3.

Example 3. *Consider the following hybrid system [50] in Figure 3, where*

$$f_1(x) = \begin{bmatrix} x_1 - x_1x_2 \\ -x_2 + x_1x_2 \end{bmatrix}, \; f_2(x) = \begin{bmatrix} x_1 + x_1^2x_2 \\ x_2 + x_1x_2 \end{bmatrix}.$$

It is required to verify that all trajectories of the system starting from the initial state I_1 in mode m_1:

$$I_1 = \{-2 \le x_1, x_2 \le -1\},$$

will never enter the unsafe region U_2 in the mode m_2:

$$U_2 = \{0 \le x_1 \le 1, -2 \le x_2 \le -1\}.$$

Table 1: Performance Evaluation

Examples	$\|X\|$	Feature	SyntheBC				nnbarrier			SOS method		
			$\mathcal{N}$	t_t	t_v	r	t_t	t_v	r	d	t	r
H1 [50]	2	$poly$	2-20-16-1	32.55	3.19	✓	-	-	×	-	-	×
H2 [50]	2	$poly$	2-20-16-1	45.37	2.89	✓	-	-	×	2	6.91	✓
H3 [47]	2	$poly, sqrt$	2-20-16-1	91.82	1.97	✓	-	-	×	2	4.69	✓
C1 [25]	2	$poly$	2-20-16-1	29.63	1.11	✓	-	-	×	-	-	×
C2 [23]	3	$poly, rational$	-	-	-	×	-	-	×	2	0.83	✓
C3 [25]	2	$poly$	2-20-16-1	61.45	4.58	✓	846.54	465.24	✓	-	-	×
C4 [4]	2	$poly, exp, sin$	2-20-16-1	46.44	10.44	✓	648.46	45.18	✓	3	1.43	✓
C5 [9]	3	$poly$	3-20-16-1	74.68	6.10	✓	2465.89	1049.68	✓	-	-	×
C6 [49]	2	$poly, exp, cos$	2-20-16-1	95.64	3.49	✓	1469.81	645.90	✓	-	-	×
C7 [18]	7	$poly$	7-20-16-1	111.92	5.16	✓	-	-	×	2	7.06	✓
C8 [7]	12	$poly, sin, cos$	12-20-16-1	292.28	4.19	✓	-	-	×	-	-	×

A neural network $\mathcal{N}$ with ReLU activation functions of the structure 2-20-16-1 is trained as the barrier certificate by SyntheBC. *The neural network barrier certificate is verified to satisfy Equation (17)-(20) and guarantees the safety of the hybrid system. For* nnbarrier *and the SOS method, they fail to synthesize the barrier certificates.*

6.2 Performance evaluation

Table 1 shows the performance evaluation of *SyntheBC*, *nnbarrier*, and the SOS method on 3 hybrid systems and 8 continuous systems. In Table 1, the example ID beginning with 'H' represents the hybrid system, and 'C' represents the continuous system; $|X|$ denotes the number of system variables; Feature denotes the type of system update functions. In the columns of SyntheBC, $\mathcal{N}$ denotes the structure of neural network barrier certificates; t_t denotes the running time of barrier certificate training process; t_v denotes the running time of barrier certificate verification process; r denotes the result flag: ✓denotes success and ×denotes failure. In the nnbarrier columns, t_t, t_v, and r are the same as those in SyntheBC columns. In the SOS method columns, d denotes the degree of the barrier certificate, t is the running time of synthesizing barrier certificate, and r is the same as that in SyntheBC columns. The running time is reported in seconds.

For the all 11 examples, *SyntheBC* verifies 10 examples, while the *nnbarrier* and the SOS method verify 4 and 5 examples, respectively. For the hybrid systems H1-3, *nnbarrier* fails to verify safety properties and the SOS method verifies two of them. For the high dimensional systems C7-8, the SOS method can verify one of them while *nnbarrier* fails to verify both.

For the 7-dimensional system C7, *nnbarrier* trains a candidate neural network barrier certificate on C7 using 6615.76s. However, the candidate barrier certificate can not be verified by the SMT-solver isat3 used by *nnbarrier* in 24 hours. To verify the safety of this candidate barrier certificate, we try to encode it to an optimization problem and solve it by Gurobi. Since the Bent-ReLU $y = 0.5x + \sqrt{0.25x^2 + 0.0001}$ which is used by *nnbarrier* as activation function is not piece-wise linear, it need to be approximated. However, due to the high complexity caused by the approximation of all hidden layer neurons with Bent-ReLU activation functions, we can not get the optimal solution in 24 hours. Thus it is reported failed.

Observing the structure of network $\mathcal{N}$ in Table 1, the sum of neural network neuron number is not greater than 50. Based on such network structures, *SyntheBC* successes to synthesize barrier certificates on all examples. Moreover, for example C3-6, the barrier certificate training time of *SyntheBC* has an order of magnitude improvement than that of *nnbarrier*. For C3, C5, and C6, the barrier certificate verification time of *SyntheBC* has more than two orders of magnitude improvement than that of *nnbarrier*. For C4, the verification time of *SyntheBC* is a quarter of that of *nnbarrier*.

The SOS method's running time is much less than that of *SyntheBC* and *nnbarrier*, since the approaches of synthesizing barrier certificate are fundamentally different. We experiment the SOS method on 11 examples and explore the degree of polynomial up to 8. However, the SOS method can only verify 5 examples and for the rest of the examples, it can not verify safety in an hour. These experimental results show that *SyntheBC* is more scalable and effective than *nnbarrier* and the SOS method.

7 CONCLUSION

In this paper, we have presented a novel method to synthesize neural networks as barrier certificates for verifying the safety properties of hybrid systems. Barrier certificates of the type neural networks are synthesized through two successive steps: network training and barrier certificates validation. The training procedure trains candidate networks from a special structure: ReLU neural networks consisting of two hidden layers. For barrier certificates validation, we propose a MIP-encoding based method to transform the problem of verifying barrier certification conditions into a group of MILP problems and a MIQCP problem. The recent advance in optimization makes the validation very effective and efficient. We have implemented the tool *SyntheBC* and evaluate its performance over a set of benchmark examples up to 12-dimensional state space. The experimental results show that our method is more scalable and effective than the classical polynomial barrier certificate method and the existing neural network based method *nnbarrier*.

ACKNOWLEDGMENTS

We gratefully acknowledge the support from the Leading-edge Technology Program of Jiangsu Natural Science Foundation No. BK20202001, the National Natural Science Foundation of China under Grant 62032010, 61772203, 61772260, Zhejiang Provincial Natural Science Foundation of China under Grant LY20F020020.

REFERENCES

[1] Hirokazu Anai and Volker Weispfenning. 2001. Reach Set Computations Using Real Quantifier Elimination. In *Proceedings of the 4th International Workshop on Hybrid Systems: Computation and Control (HSCC '01, Vol. 14)*. Springer, London, UK, 63–76.

[2] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. 2004. *Convex optimization*. Cambridge university press.

[3] George Cybenko. 1989. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems* 2, 4 (1989), 303–314.

[4] Liyun Dai, Ting Gan, Bican Xia, and Naijun Zhan. 2017. Barrier Certificates Revisited. *Journal of Symbolic Computation* 80 (2017), 62–86.

[5] Souradeep Dutta, Susmit Jha, Sriram Sankaranarayanan, and Ashish Tiwari. 2018. Output range analysis for deep feedforward neural networks. In *NASA Formal Methods Symposium*. Springer, 121–138.

[6] Chuchu Fan and Sayan Mitra. 2015. Bounded Verification with On-the-Fly Discrepancy Computation. In *Automated Technology for Verification and Analysis*, Bernd Finkbeiner, Geguang Pu, and Lijun Zhang (Eds.). Springer International Publishing, Cham, 446–463.

[7] Sicun Gao. [n.d.]. Quadcopter Model. ([n. d.]). https://github.com/dreal/benchmarks/ blob/master/inv/quadcopter_nonlinear.inv.

[8] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. *Deep learning*. Vol. 1. MIT press Cambridge.

[9] Eric Goubault, J-H Jourdan, Sylvie Putot, and Sriram Sankaranarayanan. 2014. Finding non-polynomial positive invariants and Lyapunov functions for polynomial systems through Darboux polynomials. In *Proceedings of the 2014 American Control Conference (ACC)*. IEEE, 3571–3578.

[10] Sumit Gulwani and Ashish Tiwari. 2008. Constraint-Based Approach for Analysis of Hybrid Systems. In *Proc. of the 20th International Conference on Computer Aided Verification (CAV)*. 190–203.

[11] Incorporate Gurobi Optimization. 2020. Gurobi optimizer reference manual. *URL https://www.gurobi.com* (2020).

[12] Kurt Hornik, Maxwell B. Stinchcombe, and Halbert White. 1989. Multilayer feedforward networks are universal approximators. *Neural Networks* 2, 5 (1989), 359–366.

[13] James Kapinski and Jyotirmoy Deshmukh. 2015. Discovering forward invariant sets for nonlinear dynamical systems. In *Interdisciplinary Topics in Applied Mathematics, Modeling and Computational Science*. Springer, 259–264.

[14] James Kapinski, Jyotirmoy V Deshmukh, Sriram Sankaranarayanan, and Nikos Aréchiga. 2014. Simulation-guided lyapunov analysis for hybrid dynamical systems. In *Proc. of the Hybrid Systems: Computation and Control (HSCC)*. ACM, 133–142.

[15] Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. 2017. Reluplex: An efficient SMT solver for verifying deep neural networks. In *International Conference on Computer Aided Verification*. Springer, 97–117.

[16] R Baker Kearfott and Vladik Kreinovich. 2013. *Applications of interval computations*. Vol. 3. Springer Science & Business Media.

[17] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[18] Edda Klipp, Ralf Herwig, Axel Kowald, Christoph Wierling, and Hans Lehrach. 2005. *Systems Biology in Practice: Concepts, Implementation and Application*. Wiley-Blackwell.

[19] Michal Kocvara and Michael Stingl. 2005. PENBMI user ́s guide. *Avaiable from http://www.penopt.com* (2005).

[20] Hui Kong, Fei He, Xiaoyu Song, William NN Hung, and Ming Gu. 2013. Exponential-condition-based barrier certificate generation for safety verification of hybrid systems. In *Proceedings of the 25th International Conference on Computer Aided Verification (CAV)*. Springer, 242–257.

[21] Hui Kong, Xiaoyu Song, Dong Han, Ming Gu, and Jiaguang Sun. 2014. A new barrier certificate for safety verification of hybrid systems. *Comput. J.* 57, 7 (2014), 1033–1045.

[22] Jiang Liu, Naijun Zhan, and Zhao Hengjun. 2011. Computing semi-algebraic invariants for polynomial dynamical systems. In *Proceedings of the International Conference on Embedded Software (EMSOFT)*. ACM, 97–106.

[23] Jiang Liu, Naijun Zhan, Hengjun Zhao, and Liang Zou. 2015. Abstraction of Elementary Hybrid Systems by Variable Transformation. In *Proceedings of the 20th International Symposium on Formal Methods*. 360–377.

[24] Zhou Lu, Hongming Pu, Feicheng Wang, Zhiqiang Hu, and Liwei Wang. 2017. The Expressive Power of Neural Networks: A View from the Width. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*. 6231–6239.

[25] Nadir Matringe, Arnaldo Vieira Moura, and Rachid Rebiha. 2010. Generating invariants for non-linear hybrid systems by linear algebraic methods. In *Proc. of the Static Analysis*. Springer, 373–389.

[26] Andrea Peruffo, Daniele Ahmed, and Alessandro Abate. 2020. Automated Formal Synthesis of Neural Barrier Certificates for Dynamical Models. *arXiv preprint arXiv:2007.03251* (2020).

[27] André Platzer and Edmund M. Clarke. 2009. Computing differential invariants of hybrid systems as fixedpoints. *Formal Methods in System Design* 35, 1 (2009), 98–120.

[28] Stephen Prajna and Ali Jadbabaie. 2004. Safety verification of hybrid systems using barrier certificates. In *Proceedings of the 7th International Workshop on Hybrid Systems: Computation and Control (HSCC)*. Springer, 477–492.

[29] S. Prajna, A. Jadbabaie, and G.J. Pappas. 2007. A framework for worst-case and stochastic safety verification using barrier certificates. *IEEE Trans. Automat. Control* 52, 8 (2007), 1415–1429.

[30] Maithra Raghu, Ben Poole, Jon M. Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein. 2017. On the Expressive Power of Deep Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*. 2847–2854.

[31] Enric Rodríguez-Carbonell and Ashish Tiwari. 2005. Generating Polynomial Invariants for Hybrid Systems. In *Proc. of the 8th ACM International Conference on Hybrid Systems: Computation and Control*. 590–605.

[32] Sriram Sankaranarayanan. 2010. Automatic invariant generation for hybrid systems using ideal fixed points. In *Proceedings of the 13th ACM International Conference on Hybrid Systems: Computation and Control*. ACM, 221–230.

[33] Sriram Sankaranarayanan. 2020. Reachability Analysis Using Message Passing over Tree Decompositions. In *Computer Aided Verification*, Shuvendu K. Lahiri and Chao Wang (Eds.). Springer International Publishing, Cham, 604–628.

[34] Sriram Sankaranarayanan, Henny Sipma, and Zohar Manna. 2008. Constructing invariants for hybrid systems. *Formal Methods in System Design* 32, 1 (2008), 25–55.

[35] Gagandeep Singh, Timon Gehr, Matthew Mirman, Markus Püschel, and Martin Vechev. 2018. Fast and effective robustness certification. In *Advances in Neural Information Processing Systems*. 10802–10813.

[36] Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. 2019. An abstract domain for certifying neural networks. *Proceedings of the ACM on Programming Languages* 3, POPL (2019), 1–30.

[37] Christoffer Sloth, George J Pappas, and Rafael Wisniewski. 2012. Compositional safety analysis using barrier certificates. In *Proceedings of the 15th ACM International Conference on Hybrid Systems: Computation and Control*. ACM, 15–24.

[38] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. 2012. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*. 2951–2959.

[39] Andrew Sogokon, Khalil Ghorbal, Paul B Jackson, and André Platzer. 2016. A Method for Invariant Generation for Polynomial Continuous Systems. In *Proceedings of the International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI)*. Springer, 268–288.

[40] Thomas Sturm and Ashish Tiwari. 2011. Verification and synthesis using real quantifier elimination. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation (ISSAC)*. ACM Press, 329–336.

[41] Vincent Tjeng and Russ Tedrake. 2017. Verifying neural networks with mixed integer programming. *arXiv preprint arXiv:1711.07356* (2017), 945–950.

[42] Vincent Tjeng, Kai Xiao, and Russ Tedrake. 2017. Evaluating robustness of neural networks with mixed integer programming. *arXiv preprint arXiv:1711.07356* (2017).

[43] Cumhur Erkan Tuncali, James Kapinski, Hisahiro Ito, and Jyotirmoy V. Deshmukh. 2018. Reasoning about safety of learning-enabled components in autonomous cyber-physical systems. In *Proceedings of the 55th Annual Design Automation Conference, DAC 2018, San Francisco, CA, USA, June 24-29, 2018*. ACM, 30:1–30:6.

[44] Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. 2018. Formal security analysis of neural networks using symbolic intervals. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*. 1599–1614.

[45] Felix Winterer. 2017. isat3. *URL https://projects.informatik.uni-freiburg.de/projects/isat3* (2017).

[46] Eric Wong and Zico Kolter. 2018. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*. PMLR, 5286–5295.

[47] Bai Xue, Martin Fränzle, Hengjun Zhao, Naijun Zhan, and Arvind Easwaran. 2019. Probably Approximate Safety Verification of Hybrid Dynamical Systems. In *International Conference on Formal Engineering Methods*. Springer, 236–252.

[48] Zhengfeng Yang, Wang Lin, and Min Wu. 2015. Exact Verification of Hybrid Systems Based on Bilinear SOS Representation. *ACM Transactions on Embedded Computing Systems* 14, 1 (2015), 1–19.

[49] Zhengfeng Yang, Min Wu, and Wang Lin. 2020. An efficient framework for barrier certificate generation of uncertain nonlinear hybrid systems. *Nonlinear Analysis: Hybrid Systems* 36 (2020), 100837.

[50] Xia Zeng, Wang Lin, Zhengfeng Yang, Xin Chen, and Lilei Wang. 2016. Darboux-type barrier certificates for safety verification of nonlinear hybrid systems. In *Proc. of 2016 International Conference on Embedded Software, EMSOFT 2016, Pittsburgh, Pennsylvania, USA, October 1-7, 2016*, Petru Eles and Rahul Mangharam (Eds.). ACM, 11:1–11:10.

[51] Hengjun Zhao, Xia Zeng, Taolue Chen, and Zhiming Liu. 2020. Synthesizing barrier certificates using neural networks. In *Proceedings of the 23rd International Conference on Hybrid Systems: Computation and Control*. 1–11.