

# 000 001 002 003 004 005 006 007 008 009 010 011 012 013 014 015 016 017 018 019 020 021 022 023 024 025 026 027 028 029 030 031 032 033 034 035 036 037 038 039 040 041 042 043 044 045 046 047 048 049 050 051 052 053 BINARY SPIKING NEURAL NETWORKS AS CAUSAL MODELS

Anonymous authors  
Paper under double-blind review

## ABSTRACT

In this paper, we provide a causal analysis of binary spiking neural networks (BSNNs) aimed at explaining their behaviors. We formally define a BSNN and represent its spiking activity as a binary causal model. Thanks to this causal representation, we are able to explain the output of the network by leveraging logic-based methods. In particular, we show that we can successfully use a SAT (Boolean satisfiability) solver to compute abductive explanations from this binary causal model. To illustrate our approach, we trained the BSNN on the standard MNIST dataset and applied our SAT-based method to finding abductive explanations of the network’s classifications based on pixel-level features. We also compared the found explanations against SHAP, a popular method used in the area of explainable AI to explain “black box” classifiers. We show that, unlike SHAP, our method guarantees that a found explanation does not contain completely irrelevant features.

## 1 INTRODUCTION

In recent times, interest in the study of binary artificial neural networks has grown, where binarization can occur at the level of the connection weights between the neural units, at the level of their activation function, or at both levels. In the field of AI, binarized neural networks (BNNs) were recently proposed by [Hubara et al. \(2016\)](#) and [Rastegari et al. \(2016\)](#), while in neuroscience particular attention has been paid to binary spiking neural networks (BSNNs) ([Kheradpisheh et al., 2022](#); [Lu & Sengupta, 2020](#)). The main difference between BNNs and BSNNs is mainly due to the presence of temporal dynamics in BSNNs over BNNs and to the fact that in BSNNs inputs are given sequentially in discrete time, while they are instantaneously presented to BNNs. Binarization obviously comes with a price on the size of the network parameters in relation to its learning power: a binary neural network requires a considerably higher number of neural units, compared to its non-binary counterpart, in order to achieve an acceptable level of accuracy in a given classification task after training. Nonetheless, this disadvantage is counterbalanced by an advantage in terms of logical representability and therefore explainability. Specifically, thanks to the Boolean nature of BNNs and BSNNs, one can represent their firing dynamics as binary causal models and, consequently, explain their behaviors in an efficient way using propositional logic.

The present paper is devoted to exploring this trade-off between accuracy and explainability in the context of BSNNs. We focus our analysis on BSNNs instead of BNNs since, from the causal point of view, the former are more general than the latter and we prefer to concentrate on the more general model first. To fully capture the causal structure of a BSNN, one has to model the firing activities of its neural units and to represent their causal dependencies over an extended time span. BNNs are less general since the presentation of the input is not sequential and, consequently, their dynamics and the resulting causal dependencies between the neural units do not extend over time. We represent the internal mechanism of a BSNN through a binary causal model and, thanks to this representation, we explain the BSNN’s behavior. Different notions of explanation exist in the literature including abductive ([Ignatiev et al., 2019](#)), contrastive ([Miller, 2021](#)), counterfactual ([Verma et al., 2020](#)) and alterfactual ([Mertes et al., 2024](#)) explanation. In the present paper, we rely on abductive explanation (AXp) because of its simplicity and its emphasis of minimality which is a guarantee of non-redundancy. For a set of input features to be an abductive explanation of a classification by a neural network, it has to be *minimally* sufficient to ensure the classification, i.e., where minimality means that all proper subsets of features are no longer sufficient for the classification. Thus, an

054 AXp is by definition non-redundant. More details about the use of abductive explanation in machine  
 055 learning are given in Section 2.  
 056

057 The paper is structured as follows. After having discussed the related work (Section 2), in Section  
 058 3 we illustrate the BSNN architecture as well as the learning task we considered, namely MNIST  
 059 classification, and the learning algorithm we used to train our BSNNs on the MNIST dataset. We put  
 060 special emphasis on the levels of accuracy we reached depending on the type of weight quantization  
 061 of the neural network, including binary quantization (i.e., weights range in  $\{0, 1\}$ ) and three-valued  
 062 quantization (i.e., weights range in  $\{-1, 0, 1\}$ ). In Section 4, we introduce the mathematical model of  
 063 the BSNN spiking dynamics. Then, in Section 5 we map it into a binary causal model that represents  
 064 the causal dependencies between the firing activities of the neural units over time. Thanks to its binary  
 065 nature, such causal dependencies are representable through a system of Boolean equations. Section  
 066 6 is devoted to the explanation of the BSNN behavior. Specifically, we present an algorithm that  
 067 combines the binary causal model with a SAT solver to compute abductive explanations of the BSNN  
 068 classification, where an abductive explanation is constructed from pixel-level features at a specific  
 069 time point. In Section 7, we present some experimental results on computation time for the algorithm.  
 070 Finally, in Section 8 we compare our logic-based approach relying on abductive explanation with  
 071 SHAP, a well-known method in the area of explainable AI. As far as we know this is the first attempt  
 072 i) to map a BSNN into a binary causal model and ii) to exploit the resulting Boolean representation  
 073 of the causal dependencies between its neural units for explaining its behavior through a SAT solver.  
 074

## 074 2 RELATED WORK 075

076 Binarized Neural Networks (BNNs) are a class of artificial neural networks (ANNs) that have been  
 077 studied extensively by researchers (Qin et al., 2020) in the deep learning community, especially by  
 078 Bengio et al. (2013) and Hubara et al. (2016), who provided a viable way to train these networks  
 079 using standard back-prop based optimisation methods. BNNs adopt an extreme form of quantization,  
 080 by resorting to binary weights and binary activation values. Tang et al. (2017) have shown that with  
 081 back-prop based methods, it is possible to train these binarized neural networks with reasonable, near  
 082 full precision accuracy. Moreover, Rastegari et al. (2016) have demonstrated a drastic reduction in  
 083 computation time and model size with XNOR-Nets owing to the fact that computationally expensive  
 084 multiply-accumulate methods in deep learning can be simplified to faster XNOR and pop-count  
 085 operations with binarized networks. Hence, due to the afore-mentioned reasons, BNNs have gained  
 086 immense popularity for resource constrained, low power, hardware efficient applications of AI. Binary  
 087 Spiking Neural Networks (BSNNs), the subject of the present paper, are the bio-plausible counterpart  
 088 of BNNs, that take inspiration from the spiking dynamics of biological neurons in the brain. The  
 089 most useful feature of BSNNs is the way in which they process input data in terms of spike encodings,  
 090 where spikes are binary all-or-none pulses in discrete time steps compared to their continuous valued  
 091 ANN counterparts (including BNNs). These spike encodings are very convenient, as they allow us to  
 092 use our formalism on both the pixel space and intermediate feature space. Works have been done to  
 093 train BSNNs using both temporal (Kheradpisheh et al., 2022) as well as rate coding schemes (Lu &  
 094 Sengupta, 2020).

095 Causal models are mathematical objects that have been extensively studied in AI (Pearl, 2009), in  
 096 logic (Halpern, 2000; 2016) and, more recently, in the field of explainable AI (Miller, 2021). They  
 097 play a crucial role in the domain of explainable AI given the urgent need to provide formally rigorous  
 098 causal explanations of the behavior of AI systems. A causal model is a system of structural equations  
 099 describing the causal dependencies between variables. Binary causal models (BCMs) that we use in  
 100 the present work are the subclass of causal models in which variables are assumed to Boolean. They  
 101 were defined and studied in depth in previous work (Chockler & Halpern, 2004; Aleksandrowicz  
 102 et al., 2017; Lorini, 2023; de Lima & Lorini, 2024). Given their close connection with propositional  
 103 logic, they offer the possibility to automate reasoning about causality with the aid of a SAT solver.

104 Abductive explanation (AXp), the concept of explanation on which we rely in the present work, is  
 105 widely used in the domain of explainable AI (Cooper & Marques-Silva, 2023). It is grounded on  
 106 previous theoretical work on abduction (Marquis, 1991) and relies on the notion of prime implicant  
 107 (PI). Thus, it is also called PI-explanation (Shih et al., 2018) or sufficient reason (Darwiche & Hirth,  
 108 2020). It has been extensively used in AI to explain tractable models such as monotone or linear  
 109 classifiers (Marques-Silva et al., 2020; Cooper & Marques-Silva, 2023; Audemard et al., 2020) as well

as intractable ones such as random forests (Izza & Marques-Silva, 2021) and boosted trees (Audemard et al., 2023). It was used by Shi et al. (2020) and Ignatiev et al. (2019) to explain artificial neural networks. On the one hand, Shi et al. (2020) compile binary neural networks into Ordered Binary Decision Diagrams (OBDDs) and use the latter to compute AXps of the networks’ classifications. Ignatiev et al. (2019) compute AXps of a neural network’s classification in a three-digit MNIST classification task using a MILP (Mixed Integer Linear Programming) encoding. Unlike us and Shi et al. (2020), Ignatiev et al. (2019) consider neural networks with real-valued weights. Two major novel contributions of our work compared to Ignatiev et al. (2019) and Shi et al. (2020) are the following. First and foremost, the notion of causality is crucial in our approach: we map a BSNN into a binary causal model and exploit this causal representation to explain it. There is no causality involved in Ignatiev et al. (2019) and Shi et al. (2020)’s analyses. Secondly, they do not consider BSNNs, while BSNNs are the central object of our analysis and the type of neural networks we want to explain with the help of logic and causal models.

### 3 NEURAL NETWORK ARCHITECTURE, LEARNING AND DATASET

In this section, we outline the details of the neural network models that we considered, along with the exact learning task, dataset and accuracies. The codes for the implementation are included in the supplementary material.

#### 3.1 LEARNING TASK

For our training purposes, we used the MNIST classification task for hand written digit recognition. We trained networks with a single fully connected hidden layer on both tasks, 3-digit and 10-digit MNIST classification. As we will show in Table 1 of Section 4, we could achieve very high accuracy with binary quantized networks on the 3-digit classification task. We could also achieve a high accuracy on the 10-digit classification task with three-value quantized networks with weights ranging over  $\{-1, 0, 1\}$ . In the experimental analysis of computation time for searching an explanation we will present in Section 7, we only focus on 3-digit MNIST classification with binary SNNs.

#### 3.2 SPIKE ENCODING

For our experiments, we used two different approaches to convert MNIST images into spikes. Firstly, we used a classic Poisson rate coding scheme (Prescott & Sejnowski, 2008) to convert images into spike trains in multiple time-steps and also a threshold-binariized scheme with just one time-step as presented in Table 1 of Section 4. We did not pursue temporal coding in our experiments since, as shown by Kheradpisheh et al. (2022), temporal coding requires larger time-steps for training with high accuracy. Since having more time-steps significantly increases the complexity of finding an explanation, we chose to not use temporal coding in this work. Nonetheless, the novel mapping of BSNNs into binary causal models we will present in Section 5 can be generalized to other forms of spike encodings. We used a simple Integrate and Fire (IF) model for our spiking neurons, since mapping BSNNs into binary causal models is easier in the absence of leaks.

#### 3.3 WEIGHT QUANTIZATION

As we will show in Section 5, mapping a BSNN into a binary causal model requires the network to have weights quantized either in a binary (i.e.,  $\{0, 1\}$ ) or a three-valued (i.e.,  $\{-1, 0, 1\}$ ) way. To train our networks, the weight quantization procedure that we adopted closely follows the XNOR-Net proposal by Rastegari et al. (2016), i.e., during a forward pass the network uses a binarized weight matrix  $\mathcal{B}(\bar{W})$ , while during the backward pass it retains a proxy full-precision weight matrix  $W$  for gradient calculation. Straight-through-estimator (STE) (Bengio et al., 2013) was used without any gradient clipping for our training. The following equations represent the two variants of the quantizing functions  $\mathcal{B}^{bin}$  and  $\mathcal{B}^{tern}$  we used:

$$\mathcal{B}^{bin}(W_{i,j}) = \begin{cases} 0, & \text{if } W_{i,j} = 0, \\ (sign(W_{i,j}) + 1)/2, & \text{if } W_{i,j} \neq 0, \end{cases} \quad (1)$$

$$\mathcal{B}^{tern}(W_{i,j}) = sign(W_{i,j}), \quad (2)$$

with  $W_{i,j}$  the  $(i, j)$ -coordinate of the weight matrix  $W$ .

162    3.4 TRAINING BSNNs  
 163

164    In order to train our networks through standard back-propagation based methods for supervised  
 165    learning, we employed a surrogate gradient descent approach (Neftci et al., 2019) with  $\arctan$  as  
 166    the surrogate function along with a STE for updating binary weights (Bengio et al., 2013), in a way  
 167    similar to Jang et al. (2020). We used MSE loss to train our networks, along with Adam optimizer  
 168    and L2 regularisation, with the following learning rate scheduler:

$$169 \quad LR_{epoch} = \frac{LR_0}{1 + \alpha * epoch}. \\ 170 \\ 171$$

172    4 FORMAL MODEL OF SPIKING NEURONS  
 173

174    In this section, we introduce the formal model of a general binary spiking neural network (BSNN) and  
 175    of its integrate-fire (IF) spiking dynamics. Spiking neurons have the ability to process rich temporal  
 176    dynamics in the data due to the state fullness of the neurons much like in recurrent neural networks  
 177    (RNNs). We first introduce the static architecture of a BSNN.

178    **Definition 1** (BSNN architecture). *The architecture of a BSNN is a tuple  $S = (\mathbf{I}, \mathbf{L}, \mathcal{R}, \mathcal{W}, \mathbb{D}, (\tau_X)_{X \in \mathbf{L}})$  where:*

- 180    •  $\mathbf{I}$  is a non-empty set of input (or external) neurons,  $\mathbf{L}$  is a non-empty set of internal neurons  
   such that  $\mathbf{I} \cap \mathbf{L} = \emptyset$ , and  $\mathbf{N} = \mathbf{I} \cup \mathbf{L}$  is the set of all neurons;
- 183    •  $\mathcal{R} \subseteq \mathbf{L} \times \mathbf{N}$  is a connectivity relation relating each internal neuron to its predecessors  
   (either internal or external);
- 185    •  $\mathcal{W} : \mathcal{R} \rightarrow \mathbb{D}$  is the weighing function for the connectivity relation, with  $\mathbb{D}$  a (possibly  
   infinite) set of numerical values;
- 188    •  $\tau_X$  is the firing threshold for the internal neuron  $X \in \mathbf{L}$ .

189    Given the architecture of a BSNN, we introduce the following notion of BSNN-compatible fire  
 190    spiking dynamics.

192    **Definition 2** (BSNN-compatible fire spiking dynamics). *Let  $S = (\mathbf{I}, \mathbf{L}, \mathcal{R}, \mathcal{W}, \mathbb{D}, (\tau_X)_{X \in \mathbf{L}})$  be the  
 193    architecture of a BSNN and let  $F = (\mathcal{F}_X)_{X \in \mathbf{N}}$  be a family of firing functions for  $S$ 's neurons, with  
 194     $\mathcal{F}_X : \mathbb{N} \rightarrow \{0, 1\}$ . We say that  $F$  represents a possible spiking dynamics for the BSNN  $S$  up to time  
 195     $t_{end} \geq 0$ , or simply  $F$  is  $S$ -compatible up to time  $t_{end}$ , if and only if the following condition holds  
 196    for every  $X \in \mathbf{L}$  and for every  $t \leq t_{end}$ :*

$$197 \quad \mathcal{F}_X(t) = \begin{cases} 0, & \text{if } t = 0, \\ \Theta(\mathcal{A}(X, t) - \tau_X), & \text{if } t > 0, \end{cases} \quad (3)$$

199    where

$$200 \quad \mathcal{A}(X, t) = \begin{cases} 0, & \text{if } t = 0, \\ \mathcal{A}(X, t-1) \cdot (1 - \mathcal{F}_X(t-1)) + \sum_{(X, X') \in \mathcal{R}} \mathcal{W}(X, X') \cdot \mathcal{F}_{X'}(t), & \text{if } t > 0, \end{cases}$$

203    and

$$204 \quad \Theta(x) = \begin{cases} 1, & \text{if } x \geq 0, \\ 0, & \text{otherwise.} \end{cases}$$

207    Some explanations of the previous two definitions are in order. The weighing function  $\mathcal{W}$  in Definition  
 208    1 specifies for each internal neuron  $X$  and each predecessor  $X' \in \mathcal{R}(X)$  the weight of the connection  
 209    from  $X'$  to  $X$ , with  $\mathcal{R}(X) = \{X' \in \mathbf{N} : (X, X') \in \mathcal{R}\}$ . In the general model, a weight can take  
 210    any value from the set of numerical values  $\mathbb{D}$ . In the rest of our paper we will only consider the BSNN  
 211    variants of the model with  $\mathbb{D} = \{-1, 0, +1\}$  or  $\mathbb{D} = \{0, +1\}$ . Thus, from a mathematical point of  
 212    view, BSNNs are nothing but special cases of SNNs with either Boolean or three-valued weights.

213    Note that by means of the connectivity relation  $\mathcal{R}$  we can specify the set of output neurons  $\mathbf{O}$  as the  
 214    internal neurons that have no successors, that is,

$$215 \quad \mathbf{O} = \{X \in \mathbf{L} : \forall X' \in \mathbf{L}, (X', X) \notin \mathcal{R}\}.$$

216 Definition 2 describes the possible spiking dynamics of a BSNN  $S$ . In particular, the firing function  
 217  $\mathcal{F}_X$  represents a possible dynamics of the internal neuron  $X$  in the BSNN architecture: it is the  
 218 Heaviside step function of the difference between the neuron's activation value and the spiking  
 219 threshold  $\tau_X$ . The firing activity of the input neurons does not depend on the firing activity of other  
 220 neurons, it is uniquely determined by the temporally sequential presentation of the input. This is the  
 221 reason why the condition for  $\mathcal{F}_X$  only applies to the case  $X \in \mathbf{L}$ .

222 The activation value of the internal neuron  $X$  at time  $t$  depends recursively on its value at time  $t - 1$   
 223 and a weighted sum over the incoming stimulus at time  $t$ . Therefore, to respect the recursive nature  
 224 of the activation function, we have to define that at time 0, the network is completely inactive, i.e., no  
 225 node  $X \in \mathbf{N}$  is firing at time  $t = 0$ . Moreover, the incoming stimulus gets perfectly integrated as in  
 226 an Integrate-Fire (IF) model, without any leak in the neurons. But there is a hard reset term in our  
 227 neuron model, which resets the activation value to zero every time it fires a spike.

228 The BSNN architectures we trained for the MNIST classification task we informally described in  
 229 Section 3 are specific instances of Definition 1. Specifically, each network has  $28 \times 28$  input neurons,  
 230 one neuron per pixel in the image to be classified. That is,

$$\mathbf{I} = \{\mathfrak{I}_{x,y} : 1 \leq x, y \leq 28\}.$$

232 Moreover, it has either 8, 16, 32, 64 or 128 hidden neurons in the intermediate (or hidden) layer that  
 233 are fully connected to the input neurons, that is,

$$\mathbf{H}^k = \{\mathfrak{H}_y : 1 \leq y \leq k\} \text{ with } k \in \{8, 16, 32, 64, 128\},$$

236 and

$$\forall \mathfrak{H}_z \in \mathbf{H}^k, \forall \mathfrak{I}_{x,y} \in \mathbf{I}, (\mathfrak{H}_z, \mathfrak{I}_{x,y}) \in \mathcal{R}.$$

238 Finally, it has 10 classification neurons in the output layer, one neuron for each digit to be recognized  
 239 in the general MNIST classification task that are fully connected to the hidden neurons, that is,

$$\mathbf{C} = \{\mathfrak{C}_z : 1 \leq z \leq 10\},$$

242 and

$$\forall \mathfrak{H}_z \in \mathbf{H}^k, \forall \mathfrak{C}_{z'} \in \mathbf{C}, (\mathfrak{C}_{z'}, \mathfrak{H}_z) \in \mathcal{R}.$$

244 Thus, in the BSNNs we considered the set of internal neurons is  $\mathbf{L} = \mathbf{H}^k \cup \mathbf{C}$ . Notice that in this  
 245 BSNN architecture the set of classification neurons coincides with the set of output neurons, that is,  
 246  $\mathbf{O} = \mathbf{C}$ .

247 BSNN architectures with binary weights are denoted by  $S_k^{bin}$  while those with three-valued weights  
 248 are denoted by  $S_k^{tern}$ , depending on the number  $k$  of their hidden units. We only trained and tested  
 249 12 variants of BSNN networks varying along the three dimensions: the specific spike encoding  
 250 used (Poisson vs. threshold binarized), as detailed in Section 3.2, the weight quantization used  
 251 ( $\{0, 1\}$  vs.  $\{-1, 0, 1\}$ ), and the number  $k \in \{8, 16, 32, 64, 128\}$  of hidden units. For each variant,  
 252 the value of  $\mathcal{W}(X, X')$  for each  $(X, X') \in \mathcal{R}$  was determined through learning. Specifically, we  
 253 have three networks for each of the following four cases: i) binary weights, Poisson encoding  
 254 and  $k \in \{8, 16, 32\}$ ; ii) binary weights, threshold binarized encoding and  $k \in \{8, 16, 32\}$ ; iii)  
 255 three-valued weights, Poisson encoding and  $k \in \{32, 64, 128\}$ ; iii) three-valued weights, threshold  
 256 binarized encoding and  $k \in \{32, 64, 128\}$ .

257 Table 1: Accuracies of different BSNN architectures trained on the MNIST digit classification task.  
 258

Model type	Number of hidden neurons (k)	Digits	Spike encoding	Time-steps ( $t_{end}$ )	Validation Accuracy (%)	Test Accuracy (%)
$S_k^{bin}$	32	1,5,9	Poisson	16	92.98	94.29
	16		Poisson	16	94.68	94.62
	8		Poisson	8	95.20	95.27
	32		Thresholded	1	92.47	93.63
	16		Thresholded	1	92.09	91.66
	8		Thresholded	1	91.29	93.41
$S_k^{tern}$	128	0,1,2,3,4,5,6,7,8,9	Poisson	4	92.00	92.16
	64		Poisson	4	91.82	92.03
	32		Poisson	4	90.55	91.06
	128		Thresholded	1	86.56	87.00
	64		Thresholded	1	84.97	86.10
	32		Thresholded	1	85.12	85.03

In Table 1, we have listed out the different accuracies of the BSNNs  $S_k^{bin}$  and  $S_k^{tern}$ . For the experimental results in section 7 we will stick to the choice of the binary variant which is colored gray in this table. For training BSNNs, we used the SpikingJelly library (Fang et al., 2023) in PyTorch for swift and open-source implementation.

## 5 CAUSAL MODEL

A causal model is a mathematical object describing the causal dependencies between variables. As emphasized in Section 2, it is a central concept of current analyses of causality in AI. A binary causal model (BCM) is nothing but a causal model in which variables are assumed to be Boolean. In a BCM causal information is expressed by means of Boolean expressions (*alias* propositional formulas), the set of Boolean expressions being generated inductively as follows: i) each Boolean variable  $p$  and symbol  $\perp$  (“contradiction”) are Boolean expressions; ii) if  $\omega$  is a Boolean expression, so is  $\neg\omega$  (“negation”); iii) if  $\omega_1$  and  $\omega_2$  are Boolean expressions, so is  $\omega_1 \wedge \omega_2$  (“conjunction”). Additional Boolean connectives are definable as abbreviations in the usual way:  $\top =_{def} \neg\perp$ ;  $\omega_1 \vee \omega_2 =_{def} \neg(\neg\omega_1 \wedge \neg\omega_2)$  (“disjunction”);  $\omega_1 \rightarrow \omega_2 =_{def} \neg\omega_1 \vee \omega_2$  (“implication”);  $\omega_1 \leftrightarrow \omega_2 =_{def} (\omega_1 \rightarrow \omega_2) \wedge (\omega_2 \rightarrow \omega_1)$ . In formal terms, a BCM is a triplet  $\Gamma = (\mathbf{U}, \mathbf{V}, \mathcal{E})$  where i)  $\mathbf{U}$  is a set of exogenous variables, ii)  $\mathbf{V}$  is a set of endogenous variables, iii)  $\mathcal{E}$  is a function mapping each endogenous variable  $p \in \mathbf{V}$  to a Boolean expression  $\mathcal{E}(p)$  of the form  $p \leftrightarrow \omega_p$ , where  $\omega_p$  is a Boolean expression built from  $\mathbf{U} \cup \mathbf{V}$  that does not contain  $p$ . Specifically, the Boolean expression  $p \leftrightarrow \omega_p$  stipulates that the endogenous variable  $p$  is true iff the condition  $\omega_p$  is true. It can be seen as the compact representation of a Boolean function for the endogenous variable  $p$ . From a binary causal model  $\Gamma = (\mathbf{U}, \mathbf{V}, \mathcal{E})$  it is straightforward to extract a causal graph representing the causal dependencies between the variables: the vertices of the causal graph are the variables in  $\mathbf{U} \cup \mathbf{V}$ , and we draw an edge from a variable  $q$  to an endogenous variable  $p$  if the Boolean expression  $\omega_p$  such that  $\mathcal{E}(p) = p \leftrightarrow \omega_p$  contains the variable  $q$ .

The model of the BSNN given in Definition 1 can be mapped into a BCM that represents the causal dependencies between the BSNN’s neural units over time. The idea of the mapping is simple: we assign a Boolean variable  $p_{X,t}$  to each neuron  $X$  for each time  $t$  in  $\{0, \dots, t_{end}\}$ , where  $t_{end}$  is the final time step at which the network stops receiving incoming spike train from the image currently being presented, as mentioned in Section 3.2. The variable  $p_{X,t}$  is true (resp. false) if the neuron  $X$  fires (resp. does not fire) at time  $t$ . The exogenous variables are for the input neurons, while the endogenous ones are for the internal neurons. The causal dependencies between the firing activities of the neurons are represented by the Boolean equations. Here, we only give the BCM for the variants of the BSNN with Boolean weights  $\{0, 1\}$ . Due to space restrictions, we could only include the BCM for the variants with three-valued weights  $\{-1, 0, 1\}$  in Section A.2 of the Appendix.

**Definition 3** (BCM for BSNN with Boolean weights). *Let  $S = (\mathbf{I}, \mathbf{L}, \mathcal{R}, \mathcal{W}, \{0, 1\}, (\tau_X)_{X \in \mathbf{L}})$  be the architecture of a BSNN with Boolean weights in the sense of Definition 1. The BCM for  $S$  is the triplet  $\Gamma_S = (\mathbf{U}_S, \mathbf{V}_S, \mathcal{E}_S)$  where  $\mathbf{U}_S = \bigcup_{0 \leq t \leq t_{end}} \mathbf{U}_S^t$ ,  $\mathbf{V}_S = \bigcup_{0 \leq t \leq t_{end}} \mathbf{V}_S^t$ ,  $\mathbf{U}_S^t = \{p_{X,t} : X \in \mathbf{I}\}$ ,  $\mathbf{V}_S^t = \{p_{X,t} : X \in \mathbf{L}\}$ , and  $\forall X \in \mathbf{L}$ :*

$$\mathcal{E}_S(p_{X,0}) = p_{X,0} \leftrightarrow \perp, \quad (4)$$

and for  $t > 0$ :

$$\begin{aligned} \mathcal{E}_S(p_{X,t}) = p_{X,t} \leftrightarrow & \left( \left( \neg p_{X,t-1} \rightarrow \bigvee_{\substack{\Omega \subseteq \mathcal{R}^+(X): \\ \mathcal{A}(X,t-1) + |\Omega| \geq \tau_X}} \left( \bigwedge_{X' \in \Omega} p_{X',t} \right) \right) \wedge \right. \\ & \left. \left( p_{X,t-1} \rightarrow \bigvee_{\substack{\Omega \subseteq \mathcal{R}^+(X): \\ |\Omega| \geq \tau_X}} \left( \bigwedge_{X' \in \Omega} p_{X',t} \right) \right) \right), \end{aligned} \quad (5)$$

with  $\mathcal{R}^+(X) = \{X' \in \mathbf{N} : (X, X') \in \mathcal{R} \text{ and } \mathcal{W}(X, X') = 1\}$ .

We conclude this section by showing that the spiking dynamics of a BSNN are correctly represented by its BCM. Specifically, let  $S = (\mathbf{I}, \mathbf{L}, \mathcal{R}, \mathcal{W}, \{0, 1\}, (\tau_X)_{X \in \mathbf{L}})$  be a BSNN with Boolean weights and  $\mathcal{I}$  a Boolean interpretation for the variables in  $\mathbf{U}_S \cup \mathbf{V}_S$ , i.e.,  $\mathcal{I} : \mathbf{U}_S \cup \mathbf{V}_S \rightarrow \{0, 1\}$ , such that for

every time  $t \in \{0, \dots, t_{end}\}$  and for every neuron  $X$ , the function  $\mathcal{F}_X$  assigns to time  $t$  the same value assigned by the interpretation  $\mathcal{I}$  to the corresponding variable  $p_{X,t}$ . Then, the family of firing functions  $F = (\mathcal{F}_X)_{X \in \mathbf{N}}$  is  $S$ -compatible up to time  $t_{end}$  if and only if  $\mathcal{I}$  satisfies all Boolean equations of the BCM  $\Gamma_S = (\mathbf{U}_S, \mathbf{V}_S, \mathcal{E}_S)$  for  $S$ . This correspondence between a BSNN and its BCM is formally expressed by the following Theorem 1 where, for any Boolean expression  $\omega$ ,  $\mathcal{I} \models \omega$  denotes the fact that the Boolean interpretation  $\mathcal{I}$  satisfies the Boolean expression  $\omega$ . For the readers unfamiliar with Boolean (propositional) logic, we remind that  $\mathcal{I} \models \omega$  iff  $Val(\mathcal{I}, \omega) = 1$ , where  $Val(\mathcal{I}, \omega)$  is defined inductively, as follows: i)  $Val(\mathcal{I}, p) = \mathcal{I}(p)$  for  $p \in (\mathbf{U}_S \cup \mathbf{V}_S)$ ; ii)  $Val(\mathcal{I}, \perp) = 0$ ; iii)  $Val(\mathcal{I}, \neg\omega) = 1 - Val(\mathcal{I}, \omega)$ ; iv)  $Val(\mathcal{I}, \omega_1 \wedge \omega_2) = \min(Val(\mathcal{I}, \omega_1), Val(\mathcal{I}, \omega_2))$ .

**Theorem 1.** *If  $\forall X \in \mathbf{N}, \forall t \leq t_{end}, \mathcal{I}(p_{X,t}) = \mathcal{F}_X(t)$  then*

$$(\mathcal{F}_X)_{X \in \mathbf{N}} \text{ is } S\text{-compatible up to time } t_{end} \text{ iff } \mathcal{I} \models \bigwedge_{p_{X,t} \in \mathbf{V}_S} \mathcal{E}_S(p_{X,t}).$$

The proof of the theorem is given in Appendix A.1.1.

## 6 EXPLANATION

In this section, we are going to show how to use binary causal models (BCMs) for formalizing and computing explanations in the context of the BSNN architectures we trained for the MNIST classification task. Following the literature on abductive explanation (AXp) (Ignatiev et al., 2019; Liu & Lorini, 2023), we define it to be a prime implicant that is actually true. Moreover, we define it in relation to a binary causal model. For simplicity, we assume an AXp (the *explanans*) is a term made of exogenous variables and the property to be explained (the *explanandum*) is a Boolean expression made of endogenous ones. This assumption is perfectly compatible with our application to the MNIST classification task in which we want to explain the network classification on the basis of the pixel-level features. Nonetheless, this assumption could be dropped with no repercussion, we would only need to suppose that the explanans and the explanandum are made of different variables.

Some preliminary notions are needed before defining AXp formally. We define a *term* to be a conjunction of literals in which a variable can occur at most once, a literal being a variable  $p$  or its negation  $\neg p$ . Terms are denoted by  $\lambda, \lambda', \dots$ . Given two terms  $\lambda, \lambda'$ , with a bit of abuse of notation, we write  $\lambda' \subseteq \lambda$  (resp.  $\lambda' \subset \lambda$ ) to mean that the set of literals appearing in  $\lambda'$  is a subset (resp. strict subset) of the set of literals appearing in  $\lambda$ . Given a BCM  $\Gamma = (\mathbf{U}, \mathbf{V}, \mathcal{E})$  and an arbitrary set of variables  $\mathbf{X} \subseteq \mathbf{U} \cup \mathbf{V}$ , we note  $Term_{\mathbf{X}}$  the set of terms built from  $\mathbf{X}$ .

**Definition 4** (Abductive explanation). *Let  $\Gamma = (\mathbf{U}, \mathbf{V}, \mathcal{E})$  be a BCM,  $\mathcal{I}_{\mathbf{U}} : \mathbf{U} \rightarrow \{0, 1\}$  a Boolean interpretation for its exogenous variables,  $\lambda \in Term_{\mathbf{U}}$  and  $\omega_0$  a Boolean expression built from  $\mathbf{V}$ . We say that  $\lambda$  is an abductive explanation (AXp) of  $\omega_0$  relative to  $\Gamma$  and  $\mathcal{I}_{\mathbf{U}}$  if and only if:*

- i)  $\mathcal{I}_{\mathbf{U}} \models \lambda$ ,
- ii)  $\models (\bigwedge_{p \in \mathbf{V}} \mathcal{E}(p) \wedge \lambda) \rightarrow \omega_0$ ,
- iii)  $\forall \lambda' \subset \lambda, \not\models (\bigwedge_{p \in \mathbf{V}} \mathcal{E}(p) \wedge \lambda') \rightarrow \omega_0$ ,

where, for a given Boolean expression  $\omega$  built from the set of variables  $\mathbf{U} \cup \mathbf{V}$ ,  $\models \omega$  means that  $\omega$  is valid, i.e.,  $\mathcal{I} \models \omega$  for every Boolean interpretation  $\mathcal{I} \in \{0, 1\}^{\mathbf{U} \cup \mathbf{V}}$ .

We computed explanations for the three BSNN architectures  $S_8^{bin}$ ,  $S_{16}^{bin}$  and  $S_{32}^{bin}$  after having trained them on the MNIST three-digit classification task. Specifically, given a trained BSNN  $S_k^{bin}$  with  $k \in \{8, 16, 32\}$ , an input sequence  $input : \{0, \dots, t_{end}\} \times \mathbf{I} \rightarrow \{0, 1\}$  and an observed output sequence  $output : \{0, \dots, t_{end}\} \times \mathbf{C} \rightarrow \{0, 1\}$  for this input, we computed an abductive explanation of the output at a chosen time  $t \in \{0, \dots, t_{end}\}$  using only variables for the input at time  $t$ . More precisely, we take the *explanandum* (i.e.,  $\omega_0$ ) to be the Boolean expression  $out_{S_k^{bin}, t} = \def \bigwedge_{\mathbf{C}_z \in \mathbf{C}: output(t, \mathbf{C}_z)=1} p_{\mathbf{C}_z, t} \wedge \bigwedge_{\mathbf{C}_z \in \mathbf{C}: output(t, \mathbf{C}_z)=0} \neg p_{\mathbf{C}_z, t}$  which represents the observed output of the network at time  $t$ . Then, we search an abductive explanation  $\lambda \in Term_{\mathbf{U}^t}$  of  $out_{S_k^{bin}, t}$  relative

378 to the BCM  $\Gamma_{S_k^{bin}}$  for the BSNN  $S_k^{bin}$  and to the Boolean interpretation  $\mathcal{I}_{\mathbf{U}_{S_k^{bin}}}$  encoding the input  
 379 sequence *input* (i.e.,  $\mathcal{I}_{\mathbf{U}_{S_k^{bin}}}(p_{\mathfrak{I}_{x,y},t}) = \text{input}(t, \mathfrak{I}_{x,y})$  for every  $t \in \{0, \dots, t_{end}\}$  and  $\mathfrak{I}_{x,y} \in \mathbf{I}$ ).  
 380 The latter condition guarantees that the found explanation represents a portion of the actual input  
 381 presented to the network at time  $t$ .

383 The following proposition highlights an important property of a BSNN’s abductive explanation:  
 384 any input feature/neuron being mentioned in an abductive explanation of the output has necessarily  
 385 a non-zero weight connection with the network’s hidden layer. This guarantees that an abductive  
 386 explanation does not contain completely irrelevant information. In Section 8 we will contrast this  
 387 result with the SHAP explanation method for which there is no guarantee that a found explanation  
 388 does not contain completely irrelevant information.

389 **Proposition 1.** *Let  $\lambda \in \text{Term}_{\mathbf{U}_{S_k^{bin}}^t}$  be an abductive explanation of  $\text{out}_{S_k^{bin},t}$ . Then,*

$$390 \quad \forall p_{\mathfrak{I}_{x,y},t} \subseteq \lambda, \exists \mathfrak{H}_z \in \mathbf{H}^k \text{ such that } \mathfrak{I}_{x,y} \in \mathcal{R}^+(\mathfrak{H}_z).$$

392 The proof of the proposition is given in Appendix A.1.2.  
 393

394 To compute an abductive explanation, we rely on a standard abductive explanation search algorithm,  
 395 whose pseudo code is presented in Algorithm 1. The algorithm is initialized with a complete term  
 396  $\lambda_{init}$  over the set of exogenous variables (i.e.,  $\mathbf{U}_{S_k^{bin}}$ ) which fully represents the actual input at the  
 397 chosen time  $t$ . Then, we systematically remove literals from the  $\lambda_{init}$  and check for the *validity* of  
 398 the condition (ii) in Definition 4, at each iteration.

---

399 **Algorithm 1** Computing Abductive Explanation

---

400 **Require:** Initial implicant  $\lambda_{init}$  and explanandum  $\omega_0$ , that *satisfies* condition (i) and (ii) in Definition  
 401 4 respectively  
 402 **Ensure:** Abductive explanation  $\lambda$   
 403   Set  $\lambda = \lambda_{init}$   
 404   **for**  $l \in \lambda$  **do**  
 405     **if**  $(\bigwedge_{p \in \mathbf{V}} \mathcal{E}(p) \wedge \lambda) \rightarrow \omega_0$  **then**  
 406        $\lambda \rightarrow \lambda \setminus l$   
 407     **end if**  
 408   **end for**  
 409   **return**  $\lambda$

---

410 At the end of the search algorithm we further verify the validity of condition (iii) in Definition 4 for a  
 411 *prime implicant* check of the resulting abductive explanation  $\lambda$ . Algorithm 1 has a time complexity of  
 412  $\mathcal{O}(|\mathbf{U}_{S_k^{bin}}|)$  which is the total number of exogenous variables in the model. This linear dependency  
 413 suggests that the algorithm’s performance scales directly with the number of input neurons.  
 414

415  
 416 **7 EXPERIMENTAL RESULTS**

417 In this section, we provide the experimental results on computing explanations for some of the  
 418 BSNNs listed in Table 1. We implemented AXp search Algorithm 1 using the open-source *Z3* SAT  
 419 solver, which is an efficient and flexible theorem proving system implemented in Python developed  
 420 by Microsoft Research. We computed the search time, the percentage of input features mentioned in  
 421 the found explanation along with a visualization of the explanation in pixel-space for a test image.  
 422 Table 2 shows a comprehensive overview of the SAT solver run-times and the length of the found  
 423 AXp for each BSNN of type  $S_k^{bin}$  listed in Table 1, with  $k \in \{8, 16, 32\}$ .  
 424

425  
 426 Table 2: Computational analysis for searching explanation.

427 428 429 430 431 Number of hidden neurons (k)	Mean search time (hrs)	Length of found explanation	
		(%) Total features	Mean
32	10.7	20.91	164
16	5.84	27.3	214
8	11.13	12.5	98

Even though  $S_8^{bin}$  has a smaller number of hidden units, the explanation search time is higher than in the cases of  $S_{16}^{bin}$  and  $S_{32}^{bin}$ . This is due to two facts. Firstly, the spiking thresholds after training for the variant  $S_8^{bin}$  are higher than the spiking thresholds for the variants  $S_{16}^{bin}$  and  $S_{32}^{bin}$ . Secondly, the size of the BCM for a given BSNN increases exponentially in the values of the BSNN's spiking thresholds, as evident from equation (5) in Definition 3. Figure 1 provides a visualization of the found abductive explanations of the outputs of a network of type  $S_{16}^{bin}$  at times 0 and 6 (i.e.,  $\text{out}_{S_{16}^{bin},0}$  and  $\text{out}_{S_{16}^{bin},6}$ ). This network achieved an accuracy of 94.62 % on the 3-digit MNIST classification task on the test dataset of 911 images, as illustrated in Table 1, Section 4. Note that the set of input neurons/features mentioned in the explanation is a subset of the set of input neurons/features connected to the network's hidden layer. This is in line with Proposition 1 in Section 6.

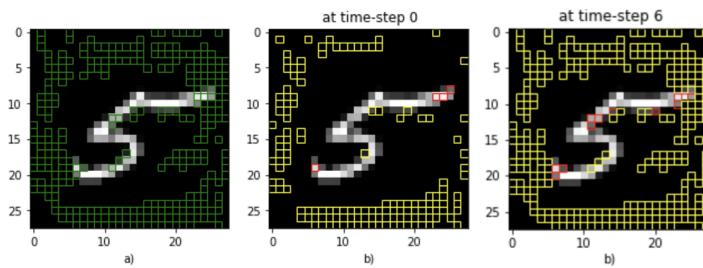


Figure 1: Image of digit 5 (a) showing in green the input neurons/features being connected with the network's hidden layer; (b) the found AXPs at times 0 and 6 showing in red the active input neurons/features (i.e., the positive literals) and in yellow the non-active input neurons/features (i.e., the negative literals) mentioned in the explanation.

## 8 COMPARISON WITH SHAP

In this section, we compare our logic-based explainability method with SHAP, a popular method widely used for interpreting predictions of machine learning models (Lundberg & Lee, 2017). For our experiments, we used the pre-existing implementation of SHAP library in Python available at <https://github.com/shap/shap>. SHAP assigns relevance scores to input features based on a sample of the input space without taking into consideration the internal dynamics of the model. Unlike our method based on causal models and abductive explanation, SHAP does not look inside the neural network and does not model the network's internal causal structure. Despite its widespread use, it has been recently shown that SHAP could assign a high relevance score to misleading or irrelevant features (Huang & Marques-Silva, 2024; 2023b;a; Letoffe et al., 2024). As discussed by Ignatiev (2020), another limitation of SHAP is that, unlike abductive explanation, it does not take minimality of an explanation into account. In Figure 2, we can see in red the positive (resp. in blue) SHAP values assigned to input pixel-level features. To compare SHAP with our method, we fixed a threshold  $\delta$  for the SHAP score and then identified the set of relevant features as those features whose SHAP score is strictly higher than  $\delta$  if positive and strictly lower than  $-\delta$  if negative.

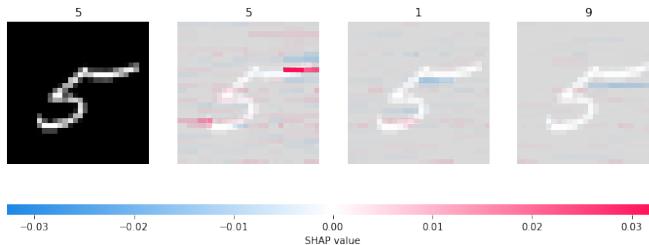


Figure 2: Visualization of SHAP relevance scores computed on a test image of digit 5.

We observed that SHAP considered relevant some input features having zero weight connections with the network’s hidden layer, which is entirely misleading. This aspect is visually represented in Figure 3. It is a consequence of the model-agnostic nature of “black box” explainability methods of which SHAP is one of the most representative examples.

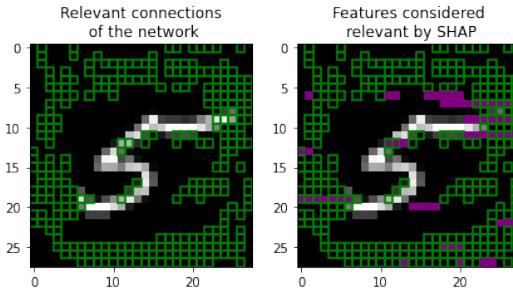


Figure 3: Green features in the two figures are those having non-zero weight connections with the network’s hidden layer. Features in purple on the right figure are considered relevant by SHAP.

Table 3 summarizes the results about the time for computing the SHAP score of an input feature and the percentage of features having zero weight connections with the hidden layer that SHAP wrongly considered relevant, depending on the size of the sample space. It turns out that on average 47% of the input features that SHAP considered relevant had zero weight connections with the network’s hidden layer. This is in stark contrast to what we demonstrated in Section 6. As Proposition 1 highlights, if we use our logic-based method, we can be sure that an explanation does not contain any input feature having zero weight connections with the network’s hidden layer.

Table 3: Percentage of features wrongly considered relevant by SHAP.

Size of the sample space	Mean computation time (s)	Features wrongly considered relevant (%)
1000000	173.6	36.95
100000	38.3	46.34
10000	4.7	57.45

As it is evident from the table 3, increasing the size of the sample space does reduce the percentage of wrongly considered features, but it comes with the cost of an increased computation time.

## 9 CONCLUSION

Let’s take stock. We have proposed a causal analysis of Binary Spiking Neural Networks (BSNNs) by mapping the models of their spiking dynamics into binary causal models (BCMs). Thanks to this mapping, we have been able to compute abductive explanations of BSNN’s decisions in the context of the MNIST classification task using a SAT-based approach. We have moreover compared our logic-based method with SHAP and highlighted the fact that, unlike SHAP, our method prevents causally irrelevant features from being mentioned in an explanation. In the current work, we only focused on the notion of abductive explanation (AXp). Future work will be devoted to extending our causal analysis of BSNNs to the notions of actual cause (Halpern & Pearl, 2005) and NESS (Necessary Element of a Sufficient Set) cause (Beckers, 2021; Halpern, 2008). Our causal framework offers the appropriate level of expressiveness to formally represent these notions and, we believe, the SAT-based approach we used for computing abductive explanations can be leveraged to compute some of these notions too. Another direction of future research is to provide a causal analysis of convolutional BSNNs (C-BSNNs) (Srinivasan & Roy, 2019) along the lines of the present work. We believe that adding convolutional layers to the network could improve accuracy in more complex datasets. Last but not least, we intend to go beyond simple visual classification tasks and leverage our logic-based causal framework to explain BSNNs trained on language datasets (Bal & Sengupta, 2024).

540 REFERENCES  
541

- 542 G. Aleksandrowicz, H. Chockler, J. Y. Halpern, and A. Ivrii. The computational complexity of  
543 structure-based causality. *Journal of Artificial Intelligence Research*, 58:431–451, 2017.
- 544 G. Audemard, F. Koriche, and P. Marquis. On tractable XAI queries based on compiled repre-  
545 sentations. In *Proceedings of the 17th International Conference on Principles of Knowledge  
546 Representation and Reasoning (KR 2020)*, pp. 838–849, 2020.
- 547 G. Audemard, S. Bellart, J.-M. Lagniez, and P. Marquis. Computing abductive explanations for  
548 boosted regression trees. In *Proceedings of the Thirty-Second International Joint Conference on  
549 Artificial Intelligence (IJCAI 2023)*, pp. 3432–3441. ijcai.org, 2023.
- 551 M. Bal and A. Sengupta. Spikingbert: Distilling BERT to train spiking language models using implicit  
552 differentiation. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence  
553 (AAAI-24)*, pp. 10998–11006. AAAI Press, 2024.
- 555 S. Beckers. The counterfactual NESS definition of causation. In *Proceedings of the Thirty-Fifth  
556 AAAI Conference on Artificial Intelligence (AAAI-21)*, pp. 6210–6217. AAAI Press, 2021.
- 557 Y. Bengio, N. Léonard, and A. Courville. Estimating or propagating gradients through stochastic  
558 neurons for conditional computation. *CoRR*, abs/1308.3432, 2013.
- 560 H. Chockler and J. Y. Halpern. Responsibility and blame: A structural-model approach. *Journal of  
561 Artificial Intelligence Research*, 22:93–115, 2004.
- 562 M. C. Cooper and J. Marques-Silva. Tractability of explaining classifier decisions. *Artificial  
563 Intelligence*, 316:103841, 2023.
- 565 A. Darwiche and A. Hirth. On the reasons behind decisions. In *Proceedings of the 24th European  
566 Conference on Artificial Intelligence (ECAI 2020)*, volume 325, pp. 712–720. IOS Press, 2020.
- 568 T. de Lima and E. Lorini. Model checking causality. In *Proceedings of the Thirty-Third International  
569 Joint Conference on Artificial Intelligence (IJCAI 2024)*. ijcai.org, 2024.
- 570 W. Fang, Y. Chen, J. Ding, Z. Yu, T. Masquelier, D. Chen, L. Huang, H. Zhou, G. Li, and Y. Tian.  
571 Spikingjelly: An open-source machine learning infrastructure platform for spike-based intelligence,  
572 2023.
- 574 J. Y. Halpern. Axiomatizing causal reasoning. *Journal of Artificial Intelligence Research*, 12:317–337,  
575 2000.
- 576 J. Y. Halpern. Defaults and normality in causal structures. In *Principles of Knowledge Representation  
577 and Reasoning: Proceedings of the Eleventh International Conference (KR 2008)*, pp. 198–208.  
578 AAAI Press, 2008.
- 580 J. Y. Halpern. *Actual causality*. MIT Press, 2016.
- 581 J. Y. Halpern and J. Pearl. Causes and explanations: a structural-model approach. Part I: Causes.  
582 *British Journal for Philosophy of Science*, 56(4):843–887, 2005.
- 584 X. Huang and J. Marques-Silva. A refutation of Shapley values for explainability. *CoRR*,  
585 abs/2309.03041, 2023a.
- 587 X. Huang and J. Marques-Silva. Refutation of Shapley values for XAI - additional evidence. *CoRR*,  
588 abs/2310.00416, 2023b.
- 589 X. Huang and J. Marques-Silva. On the failings of Shapley values for explainability. *International  
590 Journal of Approximate Reasoning*, 171:109112, 2024.
- 592 I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio. Binarized neural networks. In  
593 D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information  
Processing Systems*, volume 29. Curran Associates, Inc., 2016.

- 594 A. Ignatiev. Towards trustable explainable AI. In *Proceedings of the Twenty-Ninth International*  
595 *Joint Conference on Artificial Intelligence, IJCAI-20*, pp. 5154–5158. ijcai.org, 2020.
- 596
- 597 A. Ignatiev, N. Narodytska, and J. Marques-Silva. Abduction-based explanations for machine learning  
598 models. In *Proceedings of the Thirty-third AAAI Conference on Artificial Intelligence (AAAI-19)*,  
599 volume 33, pp. 1511–1519, 2019.
- 600 Y. Izza and J. Marques-Silva. On explaining random forests with SAT. In *Proceedings of the Thirtieth*  
601 *International Joint Conference on Artificial Intelligence (IJCAI 2021)*, pp. 2584–2591, 2021.
- 602
- 603 H. Jang, N. Skatchkovsky, and O. Simeone. Bisnn: Training spiking neural networks with binary  
604 weights via bayesian learning. *2021 IEEE Data Science and Learning Workshop (DSLW)*, pp. 1–6,  
605 2020.
- 606 S. R. Kheradpisheh, M. Mirsadeghi, and T. Masquelier. Bs4nn: Binarized spiking neural networks  
607 with temporal coding and learning. *Neural Processing Letters*, 54, 04 2022.
- 608 O. Letoffe, X. Huang, N. Asher, and J. Marques-Silva. From shap scores to feature importance scores.  
609 *CoRR*, 2024.
- 610
- 611 X. Liu and E. Lorini. A unified logical framework for explanations in classifier systems. *Journal of*  
612 *Logic and Computation*, 33(2):485–515, 2023.
- 613 E. Lorini. A rule-based modal view of causal reasoning. In *Proceedings of the Thirty-Second*  
614 *International Joint Conference on Artificial Intelligence (IJCAI 2023)*, pp. 3286–3295, 2023.
- 615
- 616 S. Lu and A. Sengupta. Exploring the connection between binary and spiking neural networks.  
617 *Frontiers in Neuroscience*, 14, June 2020. ISSN 1662-453X.
- 618 S. M. Lundberg and S. I. Lee. A unified approach to interpreting model predictions. In I. Guyon,  
619 U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.),  
620 *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
- 621
- 622 J. Marques-Silva, T. Gerspacher, M. C. Cooper, A. Ignatiev, and N. Narodytska. Explaining naive  
623 bayes and other linear classifiers with polynomial time and delay. In *Advances in Neural Information*  
624 *Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020*,  
625 2020.
- 626 P. Marquis. Extending abduction from propositional to first-order logic. In *Proceedings of the*  
627 *International Workshop on Fundamentals of Artificial Intelligence Research (FAIR'91)*, LNCS, pp.  
628 141–155. Springer, 1991.
- 629 S. Mertes, T. Huber, C. Karle, K. Weitz, R. Schlagowski, C. Conati, and E. André. Relevant  
630 irrelevance: Generating alterfactual explanations for image classifiers. In *Proceedings of the*  
631 *Thirty-Third International Joint Conference on Artificial Intelligence IJCAI 2024*, pp. 467–475.  
632 ijcai.org, 2024.
- 633 T. Miller. Contrastive explanation: a structural-model approach. *The Knowledge Engineering Review*,  
634 36, 2021.
- 635
- 636 E. O. Neftci, H. Mostafa, and F. Zenke. Surrogate gradient learning in spiking neural networks:  
637 Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal*  
638 *Processing Magazine*, 36:51–63, 2019.
- 639 J. Pearl. *Causality: Models, Reasoning and Inference*. Cambridge University Press, 2009.
- 640
- 641 S. A. Prescott and T. J. Sejnowski. Spike-rate coding and spike-time coding are affected oppositely  
642 by different adaptation mechanisms. *Journal of Neuroscience*, 28(50):13649–13661, 2008.
- 643 H. Qin, R. Gong, X. Liu, X. Bai, J. Song, and N. Sebe. Binary neural networks: A survey. *Pattern*  
644 *Recognition*, 105:107281, 2020. ISSN 0031-3203.
- 645
- 646 M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi. Xnor-net: Imagenet classification using binary  
647 convolutional neural networks. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling (eds.),  
648 *Computer Vision – ECCV 2016*, pp. 525–542, Cham, 2016. Springer International Publishing.

- 648 W. Shi, A. Shih, A. Darwiche, and A. Choi. On tractable representations of binary neural networks.  
 649 In *Proceedings of the 17th International Conference on Principles of Knowledge Representation  
 650 and Reasoning (KR 2020)*, pp. 882–892, 2020.
- 651 A. Shih, A. Choi, and A. Darwiche. A symbolic approach to explaining bayesian network classifiers.  
 652 In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence  
 653 (IJCAI 2018)*, pp. 5103–5111, 2018.
- 654 G. Srinivasan and K. Roy. Restocnet: Residual stochastic binary convolutional spiking neural  
 655 network for memory-efficient neuromorphic computing. *Frontiers in Neuroscience*, 13, 2019.  
 656 ISSN 1662-453X.
- 657 W. Tang, G. Hua, and L. Wang. How to train a compact binary neural network with high accuracy?  
 658 In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*, pp.  
 659 2625–2631. AAAI Press, 2017.
- 660 S. Verma, J. P. Dickerson, and K. Hines. Counterfactual explanations for machine learning: A review.  
 661 *CoRR*, 2020.
- 662

## A APPENDIX

In this Appendix, we present i) the proofs of the mathematical results presented in the paper and ii) the formal causal model for the BSNN architecture  $S_k^{tern}$ .

### A.1 PROOFS

#### A.1.1 PROOF OF THEOREM 1

*Proof.* ( $\Rightarrow$ ) We first prove the left-to-right direction. Suppose i)  $(\mathcal{F}_X)_{X \in \mathbf{N}}$  is  $S$ -compatible up to time  $t_{end}$  and ii)  $\forall X \in \mathbf{N}, \forall t \leq t_{end}, \mathcal{F}_X(t) = \mathcal{I}(p_{X,t})$ . We are going to prove that  $\mathcal{I} \models \mathcal{E}(p_{X,t})$  for every  $t \in \{0, \dots, t_{end}\}$  and for every  $X \in \mathbf{L}$ . The case  $t = 0$  is evident. In fact,  $\mathcal{I}(p_{X,0}) = \mathcal{F}_X(0) = 0$  by i) and ii). Moreover,  $\mathcal{I}(p_{X,0}) = 0$  iff  $Val(\mathcal{I}, p_{X,0} \leftrightarrow \perp) = 1$ , and  $Val(\mathcal{I}, p_{X,0} \leftrightarrow \perp) = 1$  iff  $\mathcal{I} \models p_{X,0} \leftrightarrow \perp$ . Thus,  $\mathcal{I} \models p_{X,0} \leftrightarrow \perp$  which is equivalent to  $\mathcal{I} \models \mathcal{E}(p_{X,0})$ . Let us prove the case  $t > 0$  by reductio ad absurdum. Suppose, toward a contradiction, that  $\mathcal{I} \not\models \mathcal{E}(p_{X,t})$ . The latter is equivalent to  $Val(\mathcal{I}, \mathcal{E}(p_{X,t})) = 0$  which is equivalent to iii)  $Val(\mathcal{I}, p_{X,t}) = 0$  and  $Val(\mathcal{I}, \chi) = 1$ , or iv)  $Val(\mathcal{I}, p_{X,t}) = 1$  and  $Val(\mathcal{I}, \chi) = 0$ , where  $\chi$  abbreviates the following Boolean expression:

$$\chi =_{def} \left( \neg p_{X,t-1} \rightarrow \bigvee_{\substack{\Omega \subseteq \mathcal{R}^+(X): \\ \mathcal{A}(X,t-1) + |\Omega| \geq \tau_X}} \left( \bigwedge_{X' \in \Omega} p_{X',t} \right) \right) \wedge \left( p_{X,t-1} \rightarrow \bigvee_{\substack{\Omega \subseteq \mathcal{R}^+(X): \\ |\Omega| \geq \tau_X}} \left( \bigwedge_{X' \in \Omega} p_{X',t} \right) \right).$$

Suppose iii) holds. On the one hand, we have  $Val(\mathcal{I}, p_{X,t}) = 0$  iff  $\mathcal{I}(p_{X,t}) = 0$ , and, by i) and ii), we have  $\mathcal{I}(p_{X,t}) = 0$  iff  $\mathcal{F}_X(t) = \Theta(\mathcal{A}(X,t) - \tau_X) = 0$ . Hence, by iii), we have  $\Theta(\mathcal{A}(X,t) - \tau_X) = 0$ . On the other hand, by ii), it is routine mathematical exercise to verify that  $Val(\mathcal{I}, \chi) = \Theta(\mathcal{A}(X,t) - \tau_X)$ . Hence, by iii), we have that  $\Theta(\mathcal{A}(X,t) - \tau_X) = 1$  which leads to a contradiction. In an analogous way we can prove that iv) leads to a contradiction.

( $\Leftarrow$ ) We are going to prove the right-to-left direction. Suppose i)  $\mathcal{I} \models \bigwedge_{p_{X,t} \in \mathbf{V}_S} \mathcal{E}_S(p_{X,t})$  and ii)  $\forall X \in \mathbf{N}, \forall t \leq t_{end}, \mathcal{F}_X(t) = \mathcal{I}(p_{X,t})$ . We are going to prove that  $(\mathcal{F}_X)_{X \in \mathbf{N}}$  is  $S$ -compatible up to time  $t_{end}$ , that is,  $\mathcal{F}_X(0) = 0$  and  $\mathcal{F}_X(t) = \Theta(\mathcal{A}(X,t) - \tau_X)$  for every  $0 < t \leq t_{end}$ . The case  $t = 0$  is evident. In fact,  $\mathcal{I}(p_{X,0}) = 0$  iff  $Val(\mathcal{I}, p_{X,0} \leftrightarrow \perp) = 1$ , and  $Val(\mathcal{I}, p_{X,0} \leftrightarrow \perp) = 1$  iff  $\mathcal{I} \models p_{X,0} \leftrightarrow \perp$ . Thus,  $\mathcal{I}(p_{X,0}) = \mathcal{F}_X(0) = 0$  by i) and ii). Let us prove the case  $0 < t \leq t_{end}$  by reductio ad absurdum. Suppose, toward a contradiction, that  $\mathcal{F}_X(t) \neq \Theta(\mathcal{A}(X,t) - \tau_X)$ . By i), we have  $\mathcal{I} \models \mathcal{E}_S(p_{X,t})$ . The latter is equivalent to  $Val(\mathcal{I}, \mathcal{E}_S(p_{X,t})) = 1$  which is equivalent to iii)  $Val(\mathcal{I}, p_{X,t}) = 1$  and  $Val(\mathcal{I}, \chi) = 1$ , or iv)  $Val(\mathcal{I}, p_{X,t}) = 0$  and  $Val(\mathcal{I}, \chi) = 0$ , where  $\chi$  is the same abbreviation as in the proof of the  $\Rightarrow$ -direction. Suppose iii) holds. On the one hand, we have  $Val(\mathcal{I}, p_{X,t}) = 1$  iff  $\mathcal{I}(p_{X,t}) = 1$ , and, by ii), we have  $\mathcal{I}(p_{X,t}) = \mathcal{F}_X(t)$ . Hence, by iii), we have  $\mathcal{F}_X(t) = 1$ . On the other hand, by ii), it is routine mathematical exercise to verify that  $Val(\mathcal{I}, \chi) = \Theta(\mathcal{A}(X,t) - \tau_X)$ . Hence, by iii), we have that  $\Theta(\mathcal{A}(X,t) - \tau_X) = 1$  and, consequently,

702  $\mathcal{F}_X(t) = 1$ . This leads to a contradiction. In an analogous way we can prove that iv) leads to a  
 703 contradiction.  $\square$

### 705 A.1.2 PROOF OF PROPOSITION 1

707 *Proof.* Suppose i) the term  $\lambda = p_{\mathfrak{I}_{x,y},t} \wedge \lambda'$  is an abductive explanation of  $\text{out}_{S_k^{bin},t}$  and, toward a  
 708 contradiction, ii)  $\exists \mathfrak{H}_z \in \mathbf{H}^k$  such that  $\mathfrak{I}_{x,y} \in \mathcal{R}^+(\mathfrak{H}_z)$ . By ii), we have that iii) for every  $p_{X,t'} \in \mathbf{V}_{S_k^{bin}}$  the Boolean equation  $\mathcal{E}(p_{X,t'})$  does not contain the variable  $p_{\mathfrak{I}_{x,y},t}$ . Moreover, by the definition  
 709 of a term and since  $p_{\mathfrak{I}_{x,y},t} \in \mathbf{U}_{S_k^{bin}}$ , iv)  $p_{\mathfrak{I}_{x,y},t}$  does not appear in  $\lambda'$  and  $p_{\mathfrak{I}_{x,y},t}$  does not appear  
 710 in  $\text{out}_{S_k^{bin},t}$ . By iii) and iv), we have that v)  $\models (\bigwedge_{p_{X,t'} \in \mathbf{V}_{S_k^{bin}}} \mathcal{E}(p_{X,t'}) \wedge p_{\mathfrak{I}_{x,y},t} \wedge \lambda') \rightarrow \text{out}_{S_k^{bin},t}$   
 711 iff  $\models (\bigwedge_{p_{X,t'} \in \mathbf{V}_{S_k^{bin}}} \mathcal{E}(p_{X,t'}) \wedge \lambda') \rightarrow \text{out}_{S_k^{bin},t}$ . Item i) implies that  $\models (\bigwedge_{p_{X,t'} \in \mathbf{V}_{S_k^{bin}}} \mathcal{E}(p_{X,t'}) \wedge$   
 712  $p_{\mathfrak{I}_{x,y},t} \wedge \lambda') \rightarrow \text{out}_{S_k^{bin},t}$  and  $\not\models (\bigwedge_{p_{X,t'} \in \mathbf{V}_{S_k^{bin}}} \mathcal{E}(p_{X,t'}) \wedge \lambda') \rightarrow \text{out}_{S_k^{bin},t}$ , which is in contradiction  
 713 with v).  $\square$

### 717 A.2 BINARY CAUSAL MODEL FOR THREE-VALUED QUANTIZATION

719 As we have already provided the formal model of the BCM corresponding to the  $S_k^{bin}$  variant of the  
 720 BSNN architecture in Section 5, we can similarly provide the BCM for  $S_k^{tern}$ .

722 The following is the binary causal model for the BSNN with three-valued weights in  $\{-1, 0, 1\}$ .

723 **Definition 5** (BCM for BSNN with three-valued weights). *Let  $S = (\mathbf{I}, \mathbf{L}, \mathcal{R}, \mathcal{W}, \{-1, 0, 1\},$   
 724  $(\tau_X)_{X \in \mathbf{L}},$ ) be the architecture of a BSNN with three-valued weights in the sense of Definition 1. The  
 725 BCM for  $S$  is the triplet  $\Gamma_S = (\mathbf{U}_S, \mathbf{V}_S, \mathcal{E}_S)$  where  $\mathbf{U}_S = \bigcup_{0 \leq t \leq t_{end}} \mathbf{U}_S^t$ ,  $\mathbf{V}_S = \bigcup_{0 \leq t \leq t_{end}} \mathbf{V}_S^t$ ,  
 726  $\mathbf{U}_S^t = \{p_{X,t} : X \in \mathbf{I}\}$ ,  $\mathbf{V}_S^t = \{p_{X,t} : X \in \mathbf{L}\}$ , and  $\forall X \in \mathbf{L}$ :*

$$\mathcal{E}_S(p_{X,0}) = p_{X,0} \leftrightarrow \perp, \quad (6)$$

729 and for  $t > 0$ :

$$\begin{aligned} \mathcal{E}_S(p_{X,t}) = p_{X,t} \leftrightarrow & \left( \left( \neg p_{X,t-1} \rightarrow \bigwedge_{\Omega \subseteq \mathcal{R}^-(X)} \left( \left( \bigwedge_{X' \in \Omega} p_{X',t} \right) \rightarrow \bigvee_{\substack{\Omega' \subseteq \mathcal{R}^+(X): \\ |\Omega'| - |\Omega| \geq \tau_X}} \left( \bigwedge_{X'' \in \Omega'} p_{X'',t} \right) \right) \right) \wedge \right. \\ & \left. \left( p_{X,t-1} \rightarrow \bigwedge_{\Omega \subseteq \mathcal{R}^-(X)} \left( \left( \bigwedge_{X' \in \Omega} p_{X',t} \right) \rightarrow \bigvee_{\substack{\Omega' \subseteq \mathcal{R}^+(X): \\ |\Omega'| - |\Omega| \geq \tau_X}} \left( \bigwedge_{X'' \in \Omega'} p_{X'',t} \right) \right) \right) \right), \quad (7) \end{aligned}$$

741 with  $\mathcal{R}^+(X) = \{X' \in \mathbf{N} : (X, X') \in \mathcal{R} \text{ and } \mathcal{W}(X, X') = 1\}$  and  $\mathcal{R}^-(X) = \{X' \in \mathbf{N} : (X, X') \in \mathcal{R} \text{ and } \mathcal{W}(X, X') = -1\}$ .

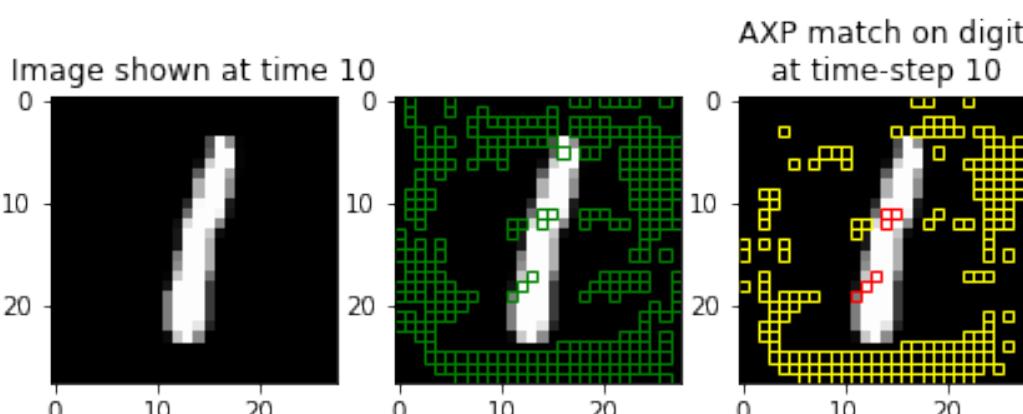
744 As we can see, the Boolean equations for the variant of BSNN with three-valued weights are  
 745 exponentially larger than the ones for the variant of BSNN with Boolean weights, we presented in  
 746 Section 5. Hence, it would be exponentially more expensive to compute abductive explanations for  
 747 the BSNN architectures of type  $S_k^{tern}$ .

### 748 A.3 FURTHER VISUAL REPRESENTATIONS OF ABDUCTIVE EXPLANATIONS

750 In Section 7 we have provided a visual representation of the abductive explanations for digit 5 (Figure  
 751 1). In the Figures 4, 5 and 6 below we provide further visualizations of the abductive explanations for  
 752 the digits 1, 9 along with another instance of digit 5 classified by the same model.

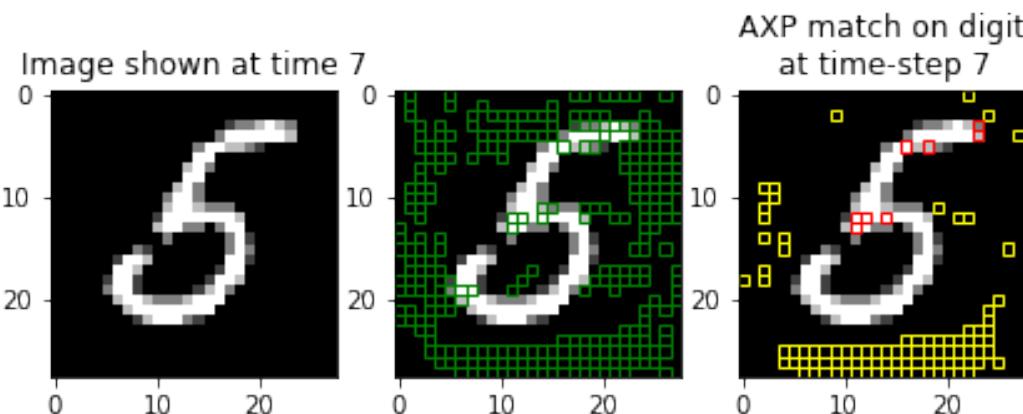
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769

Figure 4: Visualization of abductive explanation for digit 1.



770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789

Figure 5: Visualization of abductive explanation for digit 5.



790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809

Figure 6: Visualization of abductive explanation for digit 9.

