SVEBI: Towards the Interpretation and Explanation of Spiking Neural Networks

Jasper De Laet*, Hamed Behzadi-Khormouji*,
Lucas Deckers, and Jose Oramas ✉

University of Antwerp, sqIRL/IDLab, imec
*Equal Contribution

Abstract. Artificial Neural Networks have demonstrated the versatility to adapt to different problems while at the same time achieving high predictive performance. A characteristic of these models is their high-computation demands, which translates into high energy consumption. Spiking Neural Networks (SNN) have been proposed as an energy-efficient alternative with promising results in several tasks. Despite being a promising alternative that could motivate massive deployment, research on the interpretation of these models is almost non-existent. To address this problem, we propose SVEBI, a method that extracts insights on the representation encoded by SNNs, via the analysis of sparse relevant internal units that drive the decision-making process. In addition, we show the use of the relevant units as a means to justify, i.e. explain, the predictions made by an SNN for a given input. In this explanation/justification task, our experiments show SVEBI is comparable, if not superior, to existing methods.

Keywords: Spiking Neural Networks · Interpretability · Explainability.

1 Introduction

Artificial Neural Networks (ANNs) have demonstrated broad applicability, frequently matching or even surpassing human performance. ANNs show their strength in several topics including, but not limited to, speech recognition [3, 46], image/video recognition [5, 17, 31], natural language processing [21, 40] or in domains like finance [13] or healthcare [33]. However, the computational cost associated with ANNs is undeniably substantial, not only during training but also during inference and deployment. As task complexity increases, so does the complexity of the models, leading to higher computational demands in both training and real-world usage [38]. A high computational cost implies a high energy consumption which leads to limitations regarding the applicability of ANNs. This indicates that utilizing a more energy-efficient neural network design could become necessary for continuous improvement in the field of AI.

In the biological neural networks found inside the brain, information is propagated between neurons through spikes (i.e., discrete binary events). It is known that these spikes are sparse in time, which means that individual spikes encode a

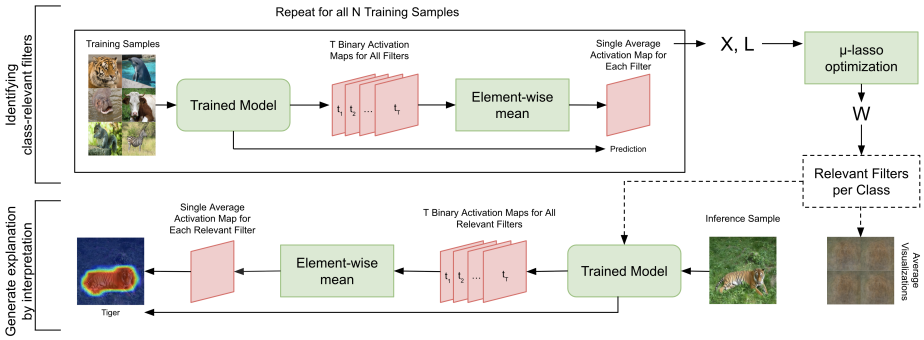

Fig. 1: Overview of the proposed methodology for the interpretation and explanation of SNNs. This includes identifying class-relevant filters, followed by the generation of average visualizations for interpretability and explanatory heatmaps.

high amount of information [2, 37]. The fact that just a few spikes are necessary to convey information throughout the brain contributes to the observation that the brain uses a low amount of energy in proportion to the tasks it can perform when compared to ANNs [2, 39, 45]. Spiking Neural Networks (SNNs) are neural networks whose design is inspired from the spiking nature of biological neural networks. SNNs use spikes to propagate information through the network over time. It has been demonstrated that SNNs, when implemented through neuromorphic hardware, can achieve significantly higher energy efficiency compared to traditional ANNs. Intel's Loihi neuromorphic chip has demonstrated energy consumption up to 16 times lower than AI models running on conventional hardware [9, 28]. Likewise, IBM's TrueNorth processor achieves a remarkable efficiency gain, using up to 100,000 times less energy per synaptic operation compared to traditional CPU/GPU-based neural networks [1, 6]. Moreover, SNNs have proven capable of achieving competitive results in various tasks, such as image classification [15, 41, 48].

As SNNs are proposed as a possible replacement for traditional ANNs, there is demand for the interpretability of SNNs, i.e. understanding its inner-workings. Understanding what an SNN learns comes down to discovering what is exactly encoded in the SNN after training. Due to the opaque characteristics of neural networks this is not an obvious task. Despite the fact that there exist numerous works dedicated to investigating the encoded features in non-spiking ANNs, a similar analysis in the context of SNNs remains limited. In this study, we endeavor to find a means for determining what a trained SNN has actually learned, i.e., the interpretation of SNNs. Towards this goal, we leverage existing well-studied interpretation methods (namely [25]) for non-spiking ANNs, while taking into account the fundamental differences between SNNs and traditional ANNs. Furthermore, comparable to the traditional ANNs, the need for justifications (explanations) towards the decisions made by a model also applies for SNNs. This introduces our secondary goal where we also aim for the explanation

of SNNs. Targeting these goals, this work proposes a new approach, SVEBI, for the interpretation as well as explanation of SNNs trained on image classification problems (see Fig. 1). In summary, we develop a method that provides insights on the representation encoded by an SNN via the analysis of filters that are critical for the performance of the classification problem addressed by the model. This method results in our two main contributions, which can be outlined as follows: (i) Considering the identified filters, we achieve interpretation of SNNs through visualizations that reveal the features that were learned by the model (ii) In addition, these identified filters are used as a basis for the generation of visual explanations in the form of heatmaps. Apart from three notable works [4, 16, 24], there has been limited direct research focused on the interpretability and explainability of SNNs.

This paper is organized as follows: Section 2 summarizes existing literature related to our work. Section 3 explains some fundamental concepts related to the contributions of this work. Then in Section 4, we present the proposed method and describe its operation. In Section 5 we validate the proposed method and discuss the corresponding results. Section 6 outlines its limitations. We provide closing remarks in Section 7.

2 Related Work

Interpretation of Neural Networks In the field of computer vision, there exist multiple interpretation methods that concern themselves with the extraction of features and concepts learned by a traditional ANN. These methods can be split up into two categories, interpretable-by-design methods and post-hoc methods. The interpretable-by-design methods (e.g., [8, 22]) are ANN models that are designed and structured in such a way that after training it is automatically possible to analyze and understand, to a certain degree, what they have actually learned. The post-hoc methods (e.g., [25, 42]) are generally methods that are applied on already trained "non-interpretable" ANN models, such that this will yield some insights regarding internal encodings of these models.

Since SNNs use spikes to propagate information through the network over time, instead of instantaneous continuous values as in traditional ANN designs, these existing interpretation methods are not directly applicable to SNNs. This motivates our objective, wherein we focus on developing an interpretation method that considers both the spiking nature and temporal dimension of SNNs. On this context, to the best of our knowledge, there is no method that provides this capability for SNNs.

Explanation Techniques for Neural Networks. In general, explanation of neural networks refers to justifying the outputs produced by the model. For image classification tasks, this often comes down to finding and investigating what information in the input was of greater importance for the model to generate a prediction for a specific input sample [27, 29, 32, 47]. Regarding this task, we focus on explanation through visualizations in the format of heatmaps. There exist model-agnostic methods like LIME [29] and RISE [27] that can be directly

applied to SNNs as they do not depend on the internal workings or structure of the model. However, methods like this require perturbing the input data where each perturbation needs to be evaluated by the model. This approach can be quite computationally expensive, which is not scalable. Moreover, their application would come at the cost of the low-energy consumption benefits of SNNs. While there are several heatmap methods available for the explanation of ANNs that are more computationally efficient, such as CAM [47] or Grad-CAM [32], directly applying these to SNNs is not a straightforward task due to the difference between the internal workings of traditional ANNs and SNNs. These methods do not take the temporal dynamics of SNNs into account. On top of this, methods like Grad-CAM require computing the gradient, which is not obvious for SNNs considering their non-differentiable nature and the use of surrogate gradients.

However, there exists a work that adapts gradient-based methods using a surrogate gradient [4]. These methods are presented in two distinct forms: one tailored for event-driven data (SNN-Grad3D and SNN-IG3D) and the other designed for 2D image classification data (SNN-Grad2D and SNN-IG2D). Following our focus on 2D image classification problems, we only consider the 2D methods in this work. SNN-Grad2D is inspired by the original saliency map from [34]. In SNN-Grad, the output attribution map represents the output class scores with respect to the input. This results in an attribution map for each timestep of the SNN. To convert this to the 2D matrix, the maps are summed over the time dimension. SNN-IG2D is based on the integrated gradients approach from [36]. This involves generating a monotonous path that gradually transforms a baseline image into the original input image in a specific number of steps. They start with a fully black image as baseline. Following this, they compute the SNN gradients with respect to the input (using surrogate gradient) for each individual step in the path, which are then integrated along the path. Here, to produce a single attribution map, the maps computed at different time-stamps are added.

Different from the above, Spike Activation Map (SAM) [16] produces heatmap visualizations per layer for each timestep. In contrast to SNN-Grad and SNN-IG, it does not need any gradients to produce its visualizations, which is advantageous because of the non-differentiable characteristics of the LIF neurons that compose SNNs. SAM is based on the temporal dynamics of SNNs to assign neuronal contribution scores in forward propagation using the history of previous spikes. It does not use any class labels, which means that the generated explanation visualization does not necessarily highlight regions of the image specific to a predicted class. Instead, the generated heatmap highlights the regions that the network focused on to generate a prediction given any image. This means that the heatmaps do not necessarily guarantee that a region of interest relevant to the predicted class is highlighted. Temporal Spike Attribution (TSA) [24] extends SAM by combining neuron spike timings, model weights, and output layer information during a forward pass to compute a final score. In contrast to SAM, TSA indirectly uses the prediction by including the membrane potentials of the output layer which define the final prediction of an SNN. However, TSA was not directly developed to be applied on convolutional neural networks, as where we

specifically focus on 2D image classification tasks. Like SAM, but in contrast to SNN-Grad2D and SNN-IG2D, we present a method that is capable of producing explanation heatmaps without the need for computing gradients. However, our method uses the predictions of the model to steer the generation of the explanations. In contrast to SAM but similar to SNN-Grad2D and SNN-IG2D, our method produces a single heatmap for all timesteps together based on the class predicted by the model, which ensures that the highlighted region is relevant to the predicted class.

3 Background

In this section, we first introduce key concepts essential for understanding the contributions made by this work (Sec. 3.1). This is followed by background information on the post-hoc interpretation method proposed in [25] since our work can be seen as an extension to the SNN domain of that method (Sec. 3.2).

3.1 Spiking Neural Networks

Artificial Neural Networks (ANNs) were originally inspired by the biological neural networks present in the brain [20, 30]. However, significant differences exist between traditional ANNs and biological neural networks in terms of structure, neural computations, learning rules, and the way information propagates between neurons [37]. These differences result in traditional ANNs consuming significantly more energy, which motivates the development of designs that align more closely with the biological counterpart. One of the most prominent examples of this are Spiking Neural Networks (SNNs), which use spikes (discrete events) to propagate information over time, emulating the spiking nature of biological neurons. A spike can be represented as a 1, and the absence of a spike as a 0. In SNNs, time is defined by timesteps T over which a single input is fed to the network. In this work, feeding a single input to an SNN means recurrently presenting the input to the network at each timestep. In our case, the input can be regarded as an image as we focus on image classification. These concepts are implemented through a spiking neuron model. Various spiking neuron models exist, but in this work the Leaky Integrate-and-Fire (LIF) model is used [18].
Leaky Integrate-and-Fire Neuron: Similar to neurons in non-spiking ANNs, LIF neurons operate by summing weighted inputs. However, instead of pushing it through an activation function, the weighted sum is added to an internal state called the membrane potential $U[t]$. When this potential reaches a threshold θ, an output spike is generated. This behaviour can be described by the Heaviside step function, where the input is the membrane potential minus the threshold value $(U[t] - \theta)$. Considering that $S_{out}[t]$ denotes the output spike value at timestep t, this function can be reformulated regarding the spiking neuron as:

$$S_{out}[t] = \begin{cases} 1, & \text{if } U[t] > \theta \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

After spiking, the membrane potential experiences a reset which can be a soft reset (subtraction by some defined value) or a hard reset (set to zero). In our case, we use a soft reset. Inherent to the LIF neuron model, the membrane potential leaks over time as well. The behaviour of the membrane potential with regard to the previous timestep can be characterized by the following discrete formula:

$$U[t] = \beta U[t-1] + WX[t] - S_{out}[t-1]\theta \tag{2}$$

where β determines the leakage and $WX[t]$ denotes the weighted input. See Fig. 2 for a schematic representation of the LIF neuron.

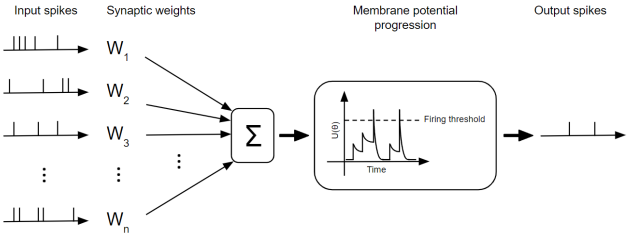

Fig. 2: Illustration of a LIF neuron. Input spikes are multiplied by their respective synaptic weights before being added to the membrane potential. The membrane potential progresses over time, where it produces a single output spike when reaching a certain threshold value. After spiking, the membrane potential experiences a reset [18]

Network structure: To form a network, multiple spiking neurons are grouped together in a similar way as with traditional ANNs. The spiking neurons are grouped together to form layers. Then, multiple layers can be stacked to produce a deep SNN.

Spike Encoding: When data is fed to an SNN, it needs to be converted into a spike representation such that the information can propagate through the network. There are several ways to do this [11], here we choose to not use an explicit input encoding technique. On the contrary, we feed our input data (continuous values) directly to the network and in this way let the network come up with its own encoding strategy in the input layer. Regarding the output of an SNN, it is necessary to convert the output of the last layer such that it represents the format of the expected output. We follow the common approach of having the number of output neurons to be equal to the number of output classes and then the neuron with the highest membrane potential after the last timestep represents the final output of the SNN [15, 19, 41]. Here, the membrane potentials of the last layer do not experience a reset after exceeding θ such that they do not emit any spikes. The loss of a prediction with regard to a target can be computed through the same loss functions used for training non-spiking ANNs. In our case, we used the cross-entropy loss.

Surrogate Gradient: At last, training SNNs can be challenging due to the non-differentiable nature of the Heaviside step function. In this work, we use the surrogate gradient method [23] to handle this. This means using the Heaviside step function in the forward pass, but replacing it by a differentiable function for computing the gradient in the backward pass. Similarly to [15], we use a sigmoidal approximation of the derivative of the step function.

3.2 ANN Interpretation and Explanation Method

When designing an interpretation method or an explanation method, it is important to base the method on attributes derived directly from the model. As explained in Sec. 3.1, the non-differentiable nature of the LIF-neuron prevents us from computing the gradients of the model. While computing gradients using a surrogate gradient function during training has shown to be effective, these are not the actual gradients of the model. This motivates our decision to adopt an activation-based approach, as the activations can be easily obtained from an SNN. In addition to this, it has been shown that SNNs following architectures and topologies inspired from non-spiking ANNs can achieve reasonable results [15, 37, 41, 44, 48]. Consequently, we opt to develop a post-hoc interpretation method, as it eliminates the need to design a new SNN architecture.

The Visual Explanation By Interpretation (VEBI) method, proposed in [25], is capable of both achieving a level of interpretation as well as explanation of ANNs. Both the interpretation and explanation are based on a post-hoc approach to identify a set of internal relevant filters per class for a given pretrained model. This method consists out of two main steps. First, the extraction of the input-wise internal responses from the network F for all N images, where N denotes the number of images used to train the model F. The second step includes using these image-wise responses to obtain a matrix $W \in R^{m \times C}$ that indicates what layer-filter pairs are relevant to what class by solving a μ-lasso problem (Equation 3). Here, m denotes the total number of considered filters (or features in general) and C the total number of classes.

$$W^* = argmin_W \|X^T W - L^T\|_F^2 \tag{3}$$

Here, X is a matrix containing the input-wise responses for all N images where $X \in R^{m \times N}$. L is a matrix containing the predicted labels (in one-hot encoding format) for all N inputs where $L \in \{0,1\}^{C \times N}$. In order to enforce that only a subset of the most relevant features are selected by W, a sparsity constraint μ is added in the following manner $\|w_j\|_1 \leq \mu, \forall j \in \{1, 2, ..., C\}$. This makes it a μ-lasso problem.

Model Interpretation: This is achieved by producing via visualizations that aim to depict the visual patterns encoded by the extracted relevant features. In [25], these visualizations are produced via average images computed from the top-k 100 images for which the identified relevant filters have the highest responses. Prior to the average operation, these images are cropped to the size

of the receptive field of the relevant filter centered at the location that has the maximum activation in the filter response.

Model Explanation: This is achieved by accompanying the class label predicted by the model with heatmap visualizations derived from the set of relevant filters supporting the prediction. Since in [25] the generation of these explanations involves computing the gradient through backpropagation — a process not directly feasible for SNNs, as outlined in Sec. 3.1—we have not further extended the explanation method presented in [25]. Instead, we have come up with our own method for generating visual explanations (Sec. 4.2).

4 Proposed Method

We propose SVEBI, a pipeline capable of both achieving interpretability and explainability of SNNs. It is composed out of two core elements. First, a method for the identification and inspection of features that are of importance to the classes of interest. This is achieved by finding the relationship between internal activations and predicted class labels (Sec. 4.1). Second, an approach for estimating layer-level responses which can serve as means for explanation by aggregating the average activation maps of filters relevant to the predicted class (Sec. 4.2). SVEBI can be considered an SNN counterpart of VEBI.

4.1 Identifying Class-Relevant Filters

We devise a method for identifying class-relevant filters by extending the method from VEBI [25] (see Sec. 3.2). Our method is an adaptation that allows for producing the W matrix that indicates the relevant layer-filter pairs per class by taking into account the disparities between non-spiking ANNs and SNNs.

The main difference lies in the first step, the extraction of input-wise responses x_i for every i^{th} training image which involves the collection of the internal responses of each filter in the architecture. For non-spiking ANNs, this means collecting a single activation map containing continuous values for each filter. However, an SNN uses spikes over time to propagate information through the network. This means that the filter responses of a single filter are activation maps consisting of binary events. Furthermore, passing a single image through the network is done over a number of timesteps T. Thus, when passing a single image through an SNN G, each filter in G produces an activation map for each timestep. We approach this difference in the following manner: When passing the i^{th} image through G, we collect T activation maps for each filter. We denote an activation map as S_{it}^{f} per filter f, where $t \in \{1, 2, 3, ..., T\}$. We squeeze the resulting T activation maps by defining a single average activation map $\bar{S_i}^{f}$ for filter f. As the activation maps can be seen as matrices, we obtain this by computing the element-wise mean as follows:

$$\bar{S_i}^{f} = \frac{1}{T} \sum_{t=1}^{T} S_{it}^{f} \tag{4}$$

This is done for each filter, followed by the subsequent steps described in VEBI [25]. Consequently, this average activation map forms the SNN equivalent for the single activation map that is collected per filter for non-spiking ANNs. As before, the above is repeated for all N training images which is followed by using the N input-wise responses to define the matrix $X \in R^{m \times N}$ and then estimate $W \in R^{m \times C}$ by solving the μ-lasso problem (see Equation 3). As described in Sec. 3.2, sparsity is enforced via the constraint μ.

4.2 Generating Explanatory Heatmaps

We generate visual explanations for SNNs in the form of heatmap visualizations using the relevant filters identified by our method (Sec. 4.1) as a basis. The heatmap visualization is based on the spiking behaviour within the identified relevant filters of a layer with respect to a single input image belonging to a specific class.

This begins by passing an image i through the SNN G and collecting the output activation map S_{it}^{f} of the specific relevant filter f for all $t \in \{1, 2, 3, ..., T\}$. Analogous to the previous sections, this is followed by producing a single average activation map $\bar{S_i}^{f}$ (via Equation 4). This is then repeated for all class-relevant filters of a single layer for the predicted class, after which we add all resulting average activation maps together. We can produce the heatmap explanation by upscaling the summed activation map to the size of the input image i, applying normalization, and then superimposing on the input image i.

4.3 Computational Considerations

The most significant computational cost introduced by our method arises during the identification of relevant filters, as this step requires processing the entire training set with the model. The generation of explanation heatmaps at inference time results in only minimal overhead, comparable to a standard forward pass through the model, as it involves collecting activation maps and executing a set of lightweight computations.

5 Evaluation

We evaluate our methods by conducting several experiments. First, we investigate the impact of the identified relevant filters on the performance of the model. This experiment serves as a fundamental sanity check that determines whether the identified filters are indeed critical for the decision-making process (Sec. 5.2). Secondly, we assess the layer-wise distribution of the identified relevant filters of the model (Sec. 5.3). This is followed by a qualitative analysis of the identified relevant filters by generating visual feedback. The objective of this experiment is to understand what the model has learned, thereby achieving model interpretation (Sec. 5.4). Then, we present our explanatory heatmaps, which we qualitatively evaluate by comparing to the maps produced by SNN-Grad2D and SNN-IG2D

(Sec. 5.5) [4]. At last, we perform a quantitative evaluation of our explanation method by using the insertion/deletion evaluation protocol [27] where we again compare to SNN-Grad2D and SNN-IG2D (Sec. 5.6).

5.1 Experimental Setup

We evaluate our methods using a shallow SNN consisting of 2 convolutional layers and 2 fully connected layers trained on MNIST [10]. From here on, we refer to this as the *shallow SNN*. In addition, we also consider a spiking equivalent of the popular CNN architecture VGG19 [35], which we refer to as *SVGG19*. For this, we consider the classification task on the AwA2 dataset [43], where the goal is to predict which animal is present in each image. AwA2 consists of 50 animal classes which comes down to 37.322 images (resized to 224x224). We used an 80/20 split for training and testing. For the shallow SNN, we trained for 5 epochs using 10 timesteps, a batch size of 100 and a learning rate of 0.3, resulting in a training accuracy of 98.96% and a best test accuracy of 98.79%. As an initial step for the SVGG19, we used the pre-trained weights provided by PyTorch [26], derived from training a non-spiking VGG19 on the ImageNet dataset [17]. Since SVGG19 shares an identical topology (in terms of layer and neuron structure) with the non-spiking VGG19, these pre-trained weights can be seamlessly transferred to SVGG19. This was followed by a training phase consisting of 50 epochs without freezing any layers, using 25 timesteps, a batch size of 8 and a learning rate of 0.003, achieving a training accuracy of 99.89% and a test accuracy of 89.83%. For both models, a spiking threshold ($\theta = 1$) is used, and the input is fed directly into the network, allowing it to autonomously develop an encoding strategy within the input layer (see Sec. 3.1). Our implementation is publicly available in our Github repository[1].

5.2 Impact of Identified Relevant Filters

To measure the impact of the identified filters on the model performance, we analyze the magnitude of the contributions of the identified filters to the model's classification accuracy. We do this by inspecting the classification results of the models when ablating the filters and comparing this to the full model performance per class. We ablate filters from a specific class and compute performance for that class, then average the results over all classes. Ablating filters is done by setting their output to zero. To verify whether the impact of a relevant filter is greater than just any filter, we compare to the classification results of the model when ablating randomly selected filters per class. The number of randomly selected filters is equal to the number of identified relevant filters. For this experiment, we do not narrow down to a specific sparsity constraint (μ), but instead identify the relevant filters for all $\mu \in \{1, 2, ..., 50\}$ and use these results for our analysis. We then repeat the above for all the values considered for the μ sparsity parameter. The average classification results over all classes of

[1] https://github.com/JasperDeLaet/SVEBI

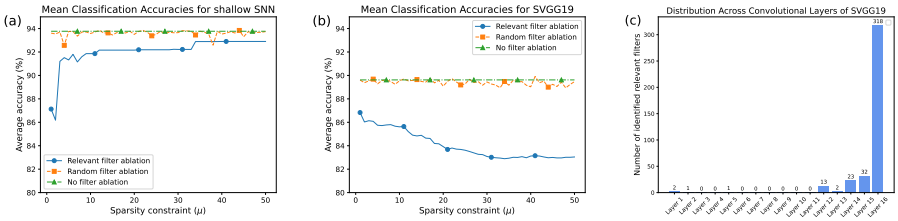

Fig. 3: (a,b) Classification results for shallow SNN and SVGG19 over different μ values, respectively. This includes the accuracy when all filters in the model are considered (green), when the output of relevant filters is ablated (blue) and when the output of random filters is ablated (orange). (c) Layer-wise distribution of the identified relevant filters over the 16 convolutional layers of the SVGG19.

this experiment can be found in Figures 3 (a,b). We observe for both models that for all μ, the accuracy when ablating relevant filters is always substantially lower than when ablating randomly selected filters. This suggests that our method is capable of identifying filters that have a greater impact on the decision-making process of the model than just any filter. The sparsity constraint μ can be considered a tunable hyperparameter, for which we use $\mu = 2$ for the shallow SNN and $\mu = 20$ for the SVGG19 in all subsequent experiments.

5.3 Layer-wise Distribution of Identified Relevant Filters

We assess the distribution of the identified relevant filters over the layers of the model. For the shallow SNN, relevant filters are found in the first convolutional layer for classes 0, 1, 4 and 6 and are present in the second convolutional layer for all classes. Figure 3 (c) shows that the identified relevant filters for the SVGG19 exhibit a skewed distribution, with a high concentration in the later layers of the model, while there are fewer relevant filters in the earlier and middle layers. This is in line with observations in the literature that state that earlier layers learn more generic features (edges, textures, and basic shapes), and that later layers focus on more complex class-specific features.

5.4 Qualitative Interpretation Feedback

Following the previous experiment, we perform a qualitative analysis in which we aim to determine what features the identified relevant filters have learned. In this way, we develop an understanding of what the model has actually learned, i.e. interpretation of the model. This qualitative analysis is performed by generating interpretation feedback in the form of average visualizations for multiple relevant filters which are resized to the input image size (Sec. 3.2). This experiment is conducted only for the SVGG19 model on the AwA2 dataset, as the low resolution of MNIST results in overly coarse visualizations that do not yield

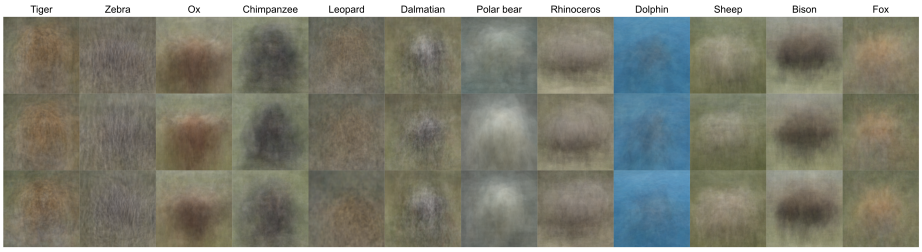

Fig. 4: Average visualizations (resized to input image size) for multiple identified relevant filters of the SVGG19 over the AwA2 dataset.

meaningful insights into the learned filter representations. Representative examples of the resulting visualizations for the identified relevant filters are shown in Figure 4. We observe that the average visualizations generally depict features that are specific to the classes for which the identified filters are relevant. For example, the average visualizations for animals with distinctive fur patterns clearly show some representation of these patterns (tiger, zebra, leopard, and dalmatian). For the ox, a head with horns can be discerned, while for the chimpanzee, a silhouette emerges. In addition to the animal-specific features, we notice that some average visualizations show characteristics of the environment in which the corresponding animal is often encountered (e.g., the polar bear or dolphin). These results generally confirm the suggestion made in Section 5.3, as the average visualizations show quite complex features.

5.5 Qualitative Assessment of Explanation Feedback

We examine the visual quality of our explanation heatmaps and compare them to the output of SNN-Grad2D and SNN-IG2D. Since SVEBI is capable of producing explanations for any identified relevant filter regardless of which layer the filter belongs to, we can explain the internal activations of the model for multiple layers. This provides deeper insights into how the model arrives at a specific prediction, leading to enhanced explainability. However, since not all layers contain relevant filters, we present SVEBI explanations for only a subset of the layers in Fig. 5. The explanations generated for the MNIST samples by SVEBI tend to highlight regions corresponding to parts of the digits. For example, the results for class 6 clearly show higher focus towards the top curve of the number 6. For the SVGG19, the explanations produced by SVEBI coming from later layers clearly show a region of interest highlighted in a manner that effectively differentiates it from the surrounding regions. Evidently, the explanations from the earlier layers show a focus on larger and less specific regions. For example, the explanations for the dolphin class sample indicate a high importance attributed to the water in the background in the first and second layers of the model. However, when considering the explanations from the final layer, it becomes evident that the model focuses on parts of the dolphin itself. Similarly for the leopard, we observe

that in layer 12 the model has an interest for patches of the fur pattern of the leopard, where in layer 16 the focus lies on the full head of the leopard.

Compared to the gradient-based explanations produced by SNN-Grad2D and SNN-IG2D, the SVEBI output is more widespread and smooth, while the gradient-based explanations are notably sparse. This trend is most clearly observed in the explanations generated for SVGG19. This aligns with existing literature, where it is commonly found that gradient-based heatmaps tend to be sparse [34, 36].

5.6 Quantitative Evaluation of Explanation Techniques

There are different approaches for quantitatively evaluating heatmap explanations, [7, 12, 14, 27]. However, we use the two automatic evaluation metrics (insertion/deletion) proposed by [27]. These evaluation metrics can be applied to a single heatmap, producing quantitative values that reflect the quality of the heatmap—specifically, whether it effectively highlights regions with high contribution to the model's prediction. In short, insertion operates by starting from a baseline image and gradually inserting pixels from the input sample based on decreasing importance determined by the heatmap. For every insertion, the performance of the model is measured. Essentially, after this procedure, the AUC of the obtained model predictions scores forms the quality score. Deletion works in a similar fashion but starts from the full input image and gradually deletes pixels from this image based on pixel importance determined by the heatmap. Again, the AUC of the model prediction scores forms the quality score. For insertion, a higher AUC score indicates better performance, as it means that the most important pixels are inserted first. Conversely, for deletion, a lower AUC score is preferred, as it suggests that the most important pixels are deleted first. In our case, the insertion baseline for AwA2 was a blurred version of the original input sample, whereas for MNIST, we used a black image (all-zero pixels). For the deletion protocol in both datasets, pixels were 'deleted' by replacing them with black pixels (i.e., setting the pixel values to zero).

To quantitatively evaluate our method and compare it to the existing SNN-Grad2D and SNN-IG2D approaches, we compute the average insertion and deletion scores over 9,920 test samples from the MNIST dataset and 7,056 test samples from the AwA2 dataset. Fig. 6 shows the performances of the models over the complete insertion and deletion procedures. Table 1 presents the corresponding AUC scores. Regarding the shallow SNN on MNIST, we observe that SVEBI outperforms the gradient-based methods in the insertion metric. For deletion, SVEBI performs comparably to SNN-Grad2D but is outperformed by SNN-IG2D. In the case of SVGG19 on AwA2, SVEBI significantly surpasses the gradient-based methods in terms of insertion scores. For deletion, SVEBI shows comparable performance, with SNN-IG2D yielding the best result. For the SVGG19-AwA2 explanations, the poor insertion results, but rather good deletion results for SNN-Grad2D and SNN-IG2D compared to SVEBI could be attributed to the sparsity of the gradient-based explanations. During the insertion procedure, this sparsity causes the model to be first presented with

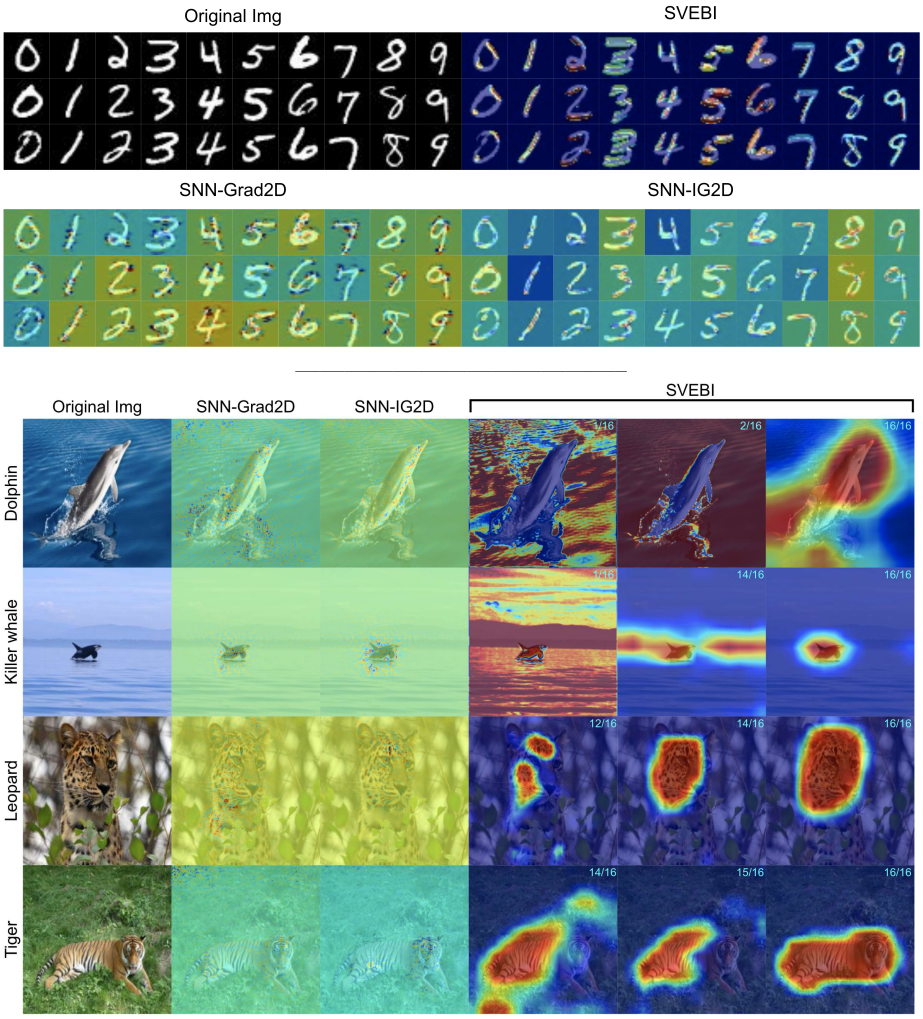

Fig. 5: Explanation heatmaps examples for a shallow SNN trained on MNIST (top) and SVGG19 trained on AwA2 (bottom). SVEBI Explanations for MNIST focus on layer 2 while those for AwA2 come from those where the relevant features/filters with highest activation are located.

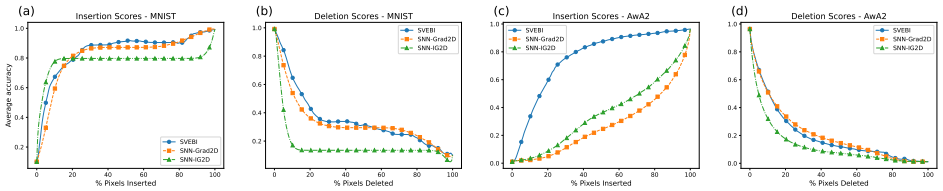

Fig. 6: Insertion/deletion scores for models trained on the MNIST (a,b) and AwA2 (c,d) datasets.

Table 1: Insertion and Deletion AUC scores over MNIST and AwA2 datasets.

Method	MNIST		AwA2	
	Insertion	Deletion	Insertion	Deletion
SNN-Grad2D	0.8199	0.3285	0.2752	0.2088
SNN-IG2D	0.7814	**0.1678**	0.3590	**0.1281**
SVEBI (ours)	**0.8459**	0.3516	**0.7582**	0.1969

important pixels that are more spread out, which lowers the model's ability to capture spatial information. This results in the model not being able to perform as well as opposed to when it is introduced with more smooth and continuous patches of pixels, which is the case for the SVEBI explanations. However, during the deletion procedure, the smoother nature of our explanation heatmaps cause the deletion of pixels to more likely focus on one region before moving to another location within the input image. Here, the sparsity of the explanations of the gradient-based methods enables a better exploration of the spatial extent of the image which results in attenuating more distinct regions. The difference between the results of MNIST and AwA2 could be attributed to the fact that for AwA2 we use a much deeper model (SVGG19) compared the shallow SNN used for MNIST. As the gradient-based methods use a surrogate gradient, the increased model depth could negatively affect the output explanations, as a deeper model requires additional operations when computing the surrogate gradients. This may increase the cumulative error arising from the use of surrogate gradients, which approximate the model's actual gradients. A more rigorous analysis is needed to validate this hypothesis, suggesting possible future research on the comparison of gradient-based and activation-based explanation methods in SNNs.

6 Limitations

For both the identification of relevant filters and the generation of explanatory heatmaps we obtain a single activation map per filter for each sample pushed through the network by performing an averaging operation over the time dimension. Here, we assume equal importance for each timestep which could cause

loss of temporal information. Possible future work could include combining this averaging operation with some temporal weighting approach. In addition, we focus solely on 2D image classification, which suggests possible future directions for adapting SVEBI to classification/regression of neuromorphic datasets.

7 Conclusion

We proposed SVEBI, a method for the analysis of the representations learned by SNNs (model interpretability) and the explainability of the outputs they produce. Our results suggest that on the interpretation side, the proposed method is capable of accurately identifying the features learned by the SNN which are important for its decision-making process. On the explainability side, quantitatively-speaking, the explanations produced by SVEBI seem to be superior to those of existing methods. On the qualitative side, the visualizations produced by our method are denser and smooth compared to the sparse heatmaps produced by previous efforts. These results show the potential of SVEBI towards assisting the study of the internal workings of existing and future SNN models. This opens the door for deeper SNN studies regarding Bias Analysis, Fairness, and other aspects that are critical for the deployment of AI technology.

Acknowledgments. This work is supported by the UAntwerp BOF DOCPRO4-NZ Project (id 41612) "Multimodal Relational Interpretation for Deep Models".

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Akopyan, F., Sawada, J., Cassidy, A., Alvarez-Icaza, R., Arthur, J., Merolla, P., Imam, N., Nakamura, Y., Datta, P., Nam, G.J., Taba, B., Beakes, M., Brezzo, B., Kuang, J.B., Manohar, R., Risk, W.P., Jackson, B., Modha, D.S.: Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **34**(10), 1537–1557 (2015)
2. Attwell, D., Laughlin, S.B.: An energy budget for signaling in the grey matter of the brain. Journal of Cerebral Blood Flow & Metabolism **21**(10), 1133–1145 (2001)
3. Bhushan, C., Kamble: Speech recognition using artificial neural network – a review. International Journal of Computing, Communication and Instrumentation Engineering **3** (2016)
4. Bitar, A., Rosales, R., Paulitsch, M.: Gradient-based feature-attribution explainability methods for spiking neural networks. Frontiers in Neuroscience **17** (2023)
5. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In: CVPR (2017)
6. Cassidy, A.S., Alvarez-Icaza, R., Akopyan, F., Sawada, J., Arthur, J.V., Merolla, P.A., Datta, P., Tallada, M.G., Taba, B., Andreopoulos, A., Amir, A., Esser, S.K., Kusnitz, J., Appuswamy, R.e.a.: Real-time scalable cortical computing at 46 giga-synaptic ops/watt with 100× speedup in time-to-solution and 100,000× reduction in energy-to-solution. In: SC (2014)

7. Chattopadhay, A., Sarkar, A., Howlader, P., Balasubramanian, V.N.: Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In: WACV (2018)
8. Chen, C., Li, O., Tao, C., Barnett, A.J., Su, J., Rudin, C.: This looks like that: Deep learning for interpretable image recognition. In: CVPR (2019)
9. Davies, M., Srinivasa, N., Lin, T.H., Chinya, G., Cao, Y., Choday, S.H., Dimou, G., Joshi, P., Imam, N., Jain, S., Liao, Y., Lin, C.K., Lines, A., Liu, R., Mathaikutty, D., McCoy, S., Paul, A., Tse, J., Venkataramanan, G., Weng, Y.H., Wild, A., Yang, Y., Wang, H.: Loihi: A neuromorphic manycore processor with on-chip learning. IEEE Micro **38**(1), 82–99 (2018)
10. Deng, L.: The mnist database of handwritten digit images for machine learning research. IEEE Signal Processing Magazine **29**(6), 141–142 (2012)
11. Eshraghian, J.K., Ward, M., Neftci, E., Wang, X., Lenz, G., Dwivedi, G., Bennamoun, M., Jeong, D.S., Lu, W.D.: Training spiking neural networks using lessons from deep learning. Proceedings of the IEEE (2023)
12. Gomez, T., Fréour, T., Mouchère, H.: Metrics for saliency map evaluation of deep learning explanation methods. In: ICPRAI (2022)
13. Huang, J., Chai, J., Cho, S.: Deep learning in finance and banking: A literature review and classification. Frontiers of Business Research in China **14** (12 2020)
14. Jung, H., Oh, Y.: Towards better explanations of class activation mapping. In: ICONIP (2023)
15. Kim, Y., Panda, P.: Revisiting batch normalization for training low-latency deep spiking neural networks from scratch. Frontiers of Neuroscience **15** (2021)
16. Kim, Y., Panda, P.: Visual explanations from spiking neural networks using inter-spike intervals. Nature Scientific Reports (2021)
17. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NeurIPS (2012)
18. Lapicque, L.: Recherches quantitatives sur l'excitation électrique des nerfs traitée comme une polarisation. J. Physiol. Pathol. Gen. **9**, 620–635 (1907)
19. Malcolm, K., Casco-Rodriguez, J.: A comprehensive review of spiking neural networks: Interpretation, optimization, efficiency, and best practices. In: arXiv:2303.10780 (2023)
20. McCulloch, W.S., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. The Bulletin of Mathematical Biophysics **5**(4), 115–133 (1943)
21. Min, B., Ross, H., Sulem, E., Veyseh, A.P.B., Nguyen, T.H., Sainz, O., Agirre, E., Heinz, I., Roth, D.: Recent advances in natural language processing via large pre-trained language models: A survey. ACM Computing Surveys **56**(2) (2023)
22. Nauta, M., van Bree, R., Seifert, C.: Neural prototype trees for interpretable fine-grained image recognition. In: CVPR (2021)
23. Neftci, E.O., Mostafa, H., Zenke, F.: Surrogate gradient learning in spiking neural networks. IEEE Signal Processing Magazine **36** (2019)
24. Nguyen, E., Nauta, M., Englebienne, G., Seifert, C.: Feature Attribution Explanations for Spiking Neural Networks . In: 2023 IEEE 5th International Conference on Cognitive Machine Intelligence. pp. 59–68 (2023)
25. Oramas, J., Wang, K., Tuytelaars, T.: Visual explanation by interpretation: Improving visual feedback capabilities of deep neural networks. In: ICLR (2019)
26. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., et al.: Pytorch: An imperative style, high-performance deep learning library. In: NeurIPS (2019)
27. Petsiuk, V., Das, A., Saenko, K.: Rise: Randomized input sampling for explanation of black-box models. In: BMVC (2018)

28. Plank, P., Rao, A., Wild, A., Maass, W.: A long short-term memory for ai applications in spike-based neuromorphic hardware. Nature Intelligence (2022)
29. Ribeiro, M.T., Singh, S., Guestrin, C.: "why should i trust you?": Explaining the predictions of any classifier. In: KDD (2016)
30. Rosenblatt, F.: The perceptron: a probabilistic model for information storage and organization in the brain. Psychological review **65 6**, 386–408 (1958)
31. Sanghvi, K., Aralkar, A., Sanghvi, S., Saha, I.: A survey on image classification techniques. SSRN Electronic Journal (01 2021)
32. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. International Journal of Computer Vision **128**(2), 336–359 (2019)
33. Shahid, N., Rappon, T., Berta, W.B.: Applications of artificial neural networks in health care organizational decision-making: A scoping review. PLoS ONE **14** (2019)
34. Simonyan, K., Vedaldi, A., Zisserman, A.: Deep inside convolutional networks: Visualising image classification models and saliency maps. In: ICLR (2014)
35. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: arXiv:1409.1556 (2014)
36. Sundararajan, M., Taly, A., Yan, Q.: Axiomatic attribution for deep networks. In: ICML (2017)
37. Tavanaei, A., Ghodrati, M., Kheradpisheh, S.R., Masquelier, T., Maida, A.S.: Deep learning in spiking neural networks. Neural Networks **111** (2019)
38. Thompson, N.C., Greenewald, K., Lee, K., Manso, G.F.: The computational limits of deep learning. In: arXiv:2007.05558 (2022)
39. Tripp, C.E., Perr-Sauer, J., Gafur, J., Nag, A., Purkayastha, A., "et al".: Measuring the energy consumption and efficiency of deep neural networks: An empirical analysis and design recommendations. In: arXiv:2403.08151 (2024)
40. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017)
41. Vicente-Sola, A., Manna, D.L., Kirkland, P., Di Caterina, G., Bihl, T.: Keys to accurate feature extraction using residual spiking neural networks. Neuromorphic Computing and Engineering **2**(4), 044001 (2022)
42. Wang, Y., Su, H., Zhang, B., Hu, X.: Interpret neural networks by extracting critical subnetworks. IEEE Transactions on Image Processing **29**, 6707–6720 (2020)
43. Xian, Y., Lampert, C.H., Schiele, B., Akata, Z.: Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. IEEE Transactions on Pattern Analysis and Machine Intelligence **41**(9), 2251–2265 (2019)
44. Yamazaki, K., Vo-Ho, V.K., Bulsara, D., Le, N.T.H.: Spiking neural networks and their applications: A review. Brain Sciences **12** (2022)
45. Zambrano, D., Nusselder, R., Scholte, H.S., Bohté, S.M.: Sparse computation in adaptive spiking neural networks. Frontiers in Neuroscience **12** (2019)
46. Zhang, Y., Chan, W., Jaitly, N.: Very deep convolutional networks for end-to-end speech recognition. In: ICASSP (2016)
47. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Learning deep features for discriminative localization (2015)
48. Zhou, C., Yu, L., Zhou, Z., Ma, Z., Zhang, H., Zhou, H., Tian, Y.: Spikingformer: Spike-driven residual learning for transformer-based spiking neural network. In: arXiv:2304.11954 (2023)