Training Deep Spiking Auto-encoders without Bursting or Dying Neurons through Regularization

Justus F. Hübotter
Donders Institute for Brain, Cognition and Behaviour
Department of Artificial Intelligence
Radboud University
the Netherlands
justus.huebotter@donders.ru.nl

Pablo Lanillos
Donders Institute for Brain, Cognition and Behaviour
Department of Artificial Intelligence
Radboud University
the Netherlands

Jakub M. Tomczak
Department of Computer Science
Vrije Universiteit Amsterdam
the Netherlands

Abstract

Spiking neural networks are a promising approach towards next-generation models of the brain in computational neuroscience. Moreover, compared to classic artificial neural networks, they could serve as an energy-efficient deployment of AI by enabling fast computation in specialized neuromorphic hardware. However, training deep spiking neural networks, especially in an unsupervised manner, is challenging and the performance of a spiking model is significantly hindered by dead or bursting neurons. Here, we apply end-to-end learning with membrane potential-based backpropagation to a spiking convolutional auto-encoder with multiple trainable layers of leaky integrate-and-fire neurons. We propose bio-inspired regularization methods to control the spike density in latent representations. In the experiments, we show that applying regularization on membrane potential and spiking output successfully avoids both dead and bursting neurons and significantly decreases the reconstruction error of the spiking auto-encoder. Training regularized networks on the MNIST dataset yields image reconstruction quality comparable to non-spiking baseline models (deterministic and variational auto-encoder) and indicates improvement upon earlier approaches. Importantly, we show that, unlike the variational auto-encoder, the spiking latent representations display structure associated with the image class.

1 Introduction

Spiking neural networks (SNNs) are biology-inspired artificial neural networks (ANNs) that have become important tools for both, artificial intelligence (AI) and computational neuroscience research [38]. In contrast to other types of ANNs, SNNs are promising candidates for scalable deployment of AI due to their low-energy, low-latency, event-driven computations in specialized
neuromorphic hardware \[^{39,34,38}\]. The internal dynamics of spiking neurons naturally include the notion of time and therefore SNNs combine key benefits from connectionism and dynamicism approaches.

Spiking neuron models, such as leaky integrate-and-fire (LIF) neurons \[^{15}\], integrate input over time and emit temporally sparse spike events. Therefore, they constitute more detailed models of biological neurons compared to standard ANNs. Generally, SNNs are applicable to the same tasks as ANNs with arbitrary (hierarchical) network architecture, such as representation learning, optimization, or closed-loop systems, and they improve the performance several orders of magnitude when temporal coding is relevant compared to non-spiking networks \[^{11}\]. Despite the new advances in neuromorphic computing \[^{15,11}\], similarly to early ANN research, SNN studies often focus on shallow feed-forward networks to solve computational benchmark tasks, such as MNIST classification—for a recent comparison see \[^{17}\]. However, deep network structures are desirable to enable hierarchical information abstraction and integration.

The auto-encoder (AE) neural model aims to learn a data-driven representation of the input information in abstracted and compressed form without performing any additional "cognitive" task on this representation (such as classification). Specifically, probabilistic latent representations of variational auto-encoders (VAEs) have recently proven useful in generative processes \[^{14,37}\], (model-based) reinforcement learning \[^{13}\] and adaptive control \[^{27}\]. While the representation of information in the latent space of non-spiking (deterministic and probabilistic) AE models is well studied \[^{26}\], how relevant findings relate to deep spiking models, however, is not always obvious. To the best of our knowledge, such deep spiking AEs (SAEs) supporting unsupervised end-to-end learning has not been properly studied in current literature. Earlier attempts to unsupervised learning in SNNs rely on layer-wise training, require additional image preprocessing, and show poor image reconstruction quality \[^{8,32}\].

Here, we aim to address this knowledge gap by applying membrane potential-based backpropagation to a convolutional SAE network with multiple trainable layers of LIF neurons for unsupervised learning. Training deep SNNs is challenging due to their non-differentiable character (i.e., LIFs produce discrete activation events) \[^{4,16,22,28,29,30}\]. Importantly, the threshold-and-reset mechanism of LIF neurons makes SNNs vulnerable to dying neurons (no activation across input stimuli) and bursting neurons (very high firing rates up to constant activation across input stimuli). This is inefficient from an information-theoretic perspective. Both, dead and constantly bursting neurons, cannot contribute actively to a differentiated representation of encoded stimuli and hinder the final performance of a model. Furthermore, a well-known computational bottleneck, in SNN implementations, appears when all neurons fire at the same time. Therefore, controlling the structure of the spiking latent representations is critical.

The goal of this paper is three-fold: (i) training a deep spiking auto-encoder end-to-end with backpropagation; (ii) avoiding undesirable neuron behavior of constant activity (bursting) and constant inactivity (dead neurons) with appropriate regularization methods; and (iii) comparing spiking model performance and latent representation against non-spiking baseline models. The spiking auto-encoder model was evaluated against two non-spiking baseline model types, namely, a deterministic AE, and a VAE on the MNIST dataset \[^{21}\], a typically used benchmark in SNN unsupervised learning. While reconstructing digits may be considered a simple task in terms of performance, the results show significant insights for understanding training and representation learning in auto-encoding SNNs. All code is available on github \[^{1}\].

\[^{1}\]https://github.com/jhuebotter/SpikingVAE

2 Methods

2.1 Problem Statement

We denote the input by \(x\) and a latent representation by \(z\). In the AE setup, \(x\) is mapped to \(z\) by an encoder network, and a reconstruction \(\hat{x}\) from \(z\) is computed by a decoder network. The encoder can be described as a function \(e\) with a set of parameters \(\theta_e\), \(z = e(x; \theta_e)\), and the decoder network is described by function \(d\) with parameters \(\theta_d\), \(\hat{x} = d(z; \theta_d)\). The auto-encoder can therefore be summarized as \(\hat{x} = d(e(x; \theta_e); \theta_d) = f(x; \theta)\). The aim of the learning process is to find values of parameters that minimize the reconstruction loss \(\theta^* = \arg\min_{\theta} \mathcal{L}(x, f(x; \theta))\), where \(\mathcal{L}_{rec} \equiv \mathcal{L}(x, \hat{x})\).
is, e.g., the $\ell_2$ distance averaged over $N$ datapoints:

$$
L_{rec} = \frac{1}{N} \sum_{n=1}^{N} (x_n - \hat{x}_n)^2.
$$

In this work, we propose to utilize spiking neural networks in the AE architecture. As a result, unlike standard AEs and VAEs, we will use binary latent variables, i.e., spikes, that will be represented by a matrix with one dimension being temporal. Since we use SNNs, we need to modify the input to the encoder (image-to-spike), and the output of the decoder (potential-to-image). The proposed architecture is schematically presented in Figure 1A.

Figure 1: **Spiking auto-encoder.** A Information processing schematic shows the difference between spiking and non-spiking models. In contrast to non-spiking models, spiking neurons integrate incoming spike events over time in a decaying membrane potential variable $u$. B Desired spiking neuron stimulus-dependent responses (green) and undesired bursting (red) or dead (blue) behaviors. Three spiking neurons, with varying weights $\theta$ (slightly shifted in the mean), are stimulated by three different digits input $\phi$. While the green neuron presents input-dependent membrane potential $u$ and sparse spiking output $\phi^\text{out}$, the red neuron displays very high firing rates across inputs and the blue neuron shows no activation across inputs. As the training uses the maximum of the cumulative membrane potential $U$ to approximate $\phi^\text{out}$ during loss calculation, it remains differentiable with respect to the network weights $\theta$. We propose regularization on $U$ and $\phi$ to balance neural responses and to obtain the desired response (green).

Controlling the spike activity patterns in SNNs, besides being important for unsupervised learning, is major to achieve the best performance when deploying in neuromorphic hardware. On the one hand, sparse representations with respect to its effects and underlying mechanisms is a desirable property [1, 9, 10, 12, 24, 36]. On the other hand, if some neurons are never or constantly active, the model capacity drops significantly. Figure 1B describes two common issues that cripple training and decrease the final performance of SNNs: dying and bursting neurons.

**Dying neurons** The dying neuron problem refers to neurons that are consistently silent or inactive independent of the input. This is also a common issue in ANNs with ReLU activation function [25]. During weight initialization or update, it regularly happens that certain neurons no longer receive sufficient input to elicit output from the activation function. In this case, and in the absence of spontaneous neuronal activity, the gradients for any following error backpropagation calculation for such neurons are 0, and no learning occurs. This essentially renders the neuron permanently inactive, or “dead”—See Figure 1B blue neuron. In some cases, this is desired, and dead neurons can be pruned to reduce network size during or after learning [7]. Alternatively, various methods have been proposed to prevent the death of too many or even any neurons, e.g. by adjusting input weights or firing thresholds dynamically [23] or neural rejuvenation [35].

**Bursting neurons** In contrast to dead neurons, the bursting neuron problem is given by the alternative extreme behavior of spiking neurons through constant activity—See Figure 1B red neuron.
Although some input is required to elicit a response in LIF neurons due to the lack of a bias term, bursting behavior may occur across inputs and therefore independent of the stimulus type.

Arguably, both dead and bursting neurons no longer participate in the representation, encoding, or decoding of information in the network, and when this occurs at a large scale, it can significantly change the network’s information storage and processing capacity. To avoid dying and bursting neurons, we propose (loosely) biologically inspired regularization methods as additional loss terms in order to leverage the already applied gradient-based weight update mechanism. The considered additional error terms are posing a soft constraint on network behavior rather than a hard limit and are described in more detail in subsection 2.3. Before that, we outline the main components of the SAE, such as a LIF neuron model, the image-to-spike encoder and the potential-to-image decoder.

2.2 Spiking auto-encoder

Leaky Integrate and Fire Model  The most influential and widely used model of neuron dynamics is the LIF model \[15, 33, 41\]. It integrates information over time in a leaky membrane-potential-inspired variable and produces binary spike events as output via a non-linear threshold-and-reset mechanism. The membrane potential dynamics and nonlinear activation function of the LIF neuron model are described by a set of equations solved at discrete time steps \(t \in [1, T]\). This is more computationally expensive than an event-based computation, but allows for the modeling of the sub-threshold dynamics of each neuron’s membrane potential as opposed to only their time of spiking. The membrane potential \(u(t)\) of any LIF neuron \(i\) is recursively dependent on itself (after potential reset) as well as incoming spike events from all projecting neurons \(j\) weighted by synaptic strength parameters \(\theta_{ij}\) as:

\[
u_i(t) = \tau u_i'(t - 1) + \sum_j \theta_{ij} \phi_j(t),
\]

where \(\tau \in [0, 1]\) is a constant hyper-parameter for decay, \(\theta_{ij}\) is the weight or synaptic strength of the connection between neuron \(i\) and \(j\). Note that no bias term is used in this model, \(\phi_j(t)\) is the binary output of neuron \(j\) and depends on the membrane potential \(u_j(t)\) and a threshold parameter \(\omega = 1\):

\[
\phi_j(t) = \begin{cases} 1, & \text{if } u_j(t) \geq \omega, \\ 0, & \text{otherwise}. \end{cases}
\]

Finally, if a neuron \(i\) emits a binary spike event of \(\phi_i(t) = 1\), the membrane potential is reset to its initial and resting potential \(u_i'(t = 0) = u_{rest} = 0\) by:

\[
u_i'(t) = \begin{cases} u_{rest}, & \text{if } u_i(t) \geq \omega, \\ u_i(t), & \text{if } \omega > u_i(t) \geq -\omega, \\ -\omega, & \text{if } u_i(t) < -\omega. \end{cases}
\]

The lower bound is set on the membrane potential \(u_i(t)\) to avoid strongly negative values, which cause undesired network behavior.

Information encoding  Before passing any input image \(x\) through a model, the image is corrupted by adding uniformly distributed noise to each pixel according to:

\[
x'_{ij} = (1 - \epsilon)x_{ij} + \epsilon \xi,
\]

with \(\epsilon \in [0, 1]\) being a hyper-parameter and \(\xi \sim \mathcal{U}(0, 1)\).

To encode the static input into a temporal sequence, we use Poisson encoding\[18, 32\] where the static input is transformed into a series of binary events for each input feature, where the density of spikes \(1s\) is approximately proportional to the magnitude of the respective input feature. This can be simulated through a stochastic process where:

\[
x''_{ij}(t) = \begin{cases} 1, & \text{if } sx'_{ij} > r(t), \\ 0, & \text{otherwise}. \end{cases}
\]

with \(r(t) \sim \mathcal{U}(0, 1)\) for every discrete time step \(t \in [1, T]\) and \(s\) being a constant scaling factor.

\[\text{Poisson encoding is one of the most common approaches along with current and time-to-first-spike encoding.}\]
**Information decoding** The outputs of the SAE model are temporal sequences of spike events \( \phi_i(t) \) as well as cumulative leaky membrane potentials \( U_i(t) \), as an approximation for the respective neurons firing rate. The latter is defined as:

\[
U_i(t) = \tau U_i(t-1) + u_i(t) .
\]  

To obtain the reconstructed \( \hat{x} \), we utilize the maximum value of \( U_i(t) \):

\[
\hat{x} = \frac{1}{s} \max_{t \in [t_{\text{min}}, T]} U_i(t) ,
\]

with \( s \) being a constant scaling factor hyper-parameter. This bounds \( \hat{x}_{ij} \in [0, \infty) \) just like the ReLU activation function and does not demand extended network activity to generate valid output. Moreover, it allows to give the network an initialization period by setting \( t_{\text{min}} > 0 \). For this work, we set that \( t_{\text{min}} = 0 \). This cumulative membrane potential can be used for training. However, using the maximum \( \max U \) limits precise time-of-spike coding [43].

**Training with membrane-potential backpropagation** The threshold or step activation function has a gradient of 0 with respect to the weights, thus, it does not allow for backpropagation. Instead, the decaying cumulative membrane potential \( U(t) \) of the output layer is used as a proxy for neural activity. During backpropagation, the threshold-and-reset mechanism is considered noise on the potential variable \( u(t) \) and is ignored by a straight-through estimate [5] as proposed for supervised learning implementations of spike-based backpropagation [22, 42].

### 2.3 Regularization methods

Here, we avoid dying and bursting neurons using soft constraints via regularization [10, ?]. The proposed regularizers for the SAE are conceptually based on different metabolic costs of processes within biological neurons to maintain the membrane potential or generate action potentials [2]. These can be categorized as regularization of weights, neuron potential, and neuron activity.

**Weight regularization** We penalize for synaptic connections of nonzero strength. This method represents a metabolic upkeep cost for each synaptic connection:

\[
\mathcal{L}_{L2} = \sum_{l=1}^{L} \sum_{i} \sum_{j} \theta_{ij}^2 .
\]

where all model parameters \( \theta \) are squared and summed for all layers \( l \in [1, L] \). The respective gradient is directly dependent on the value of the synaptic strength. This regularization method is commonly applied in statistical models and machine learning and yields generally smaller parameter values with a focus on penalizing specifically largely positive or negative values. It is conceivable that in biological neurons this mechanism has a physiological counterpart, in the sense that the presence of a synapse comes with metabolic upkeep cost, which increases depending on the strength of the synaptic connection. This concept of a metabolic cost extends to neuron behavior and is the basis for the following regularization terms.

**Potential regularization** We penalize for membrane potential deviating from the resting state, and is therefore specific to SNN models. In this context, however, it fulfills an immensely important function, as it prevents neurons getting trapped in a state of constant inhibition. In SNNs the problem of dying neurons can be avoided by directly penalizing the membrane potential deviation from resting state, independent of the respective neuron’s spiking output. Mathematically, this is defined as the two terms:

\[
\mathcal{L}_{P1} = \sum_{l=1}^{L} \sum_{i} |U_{li}(t_{\text{max}})| \quad \text{and} \quad \mathcal{L}_{P2} = \sum_{l=1}^{L} \sum_{i} U_{li}(t_{\text{max}})^2 .
\]

The underlying assumption is that there is a mechanism available, which allows a biological neuron to sense and keep track of recent membrane potential deviation trends and to adjust its sensitivity to excitation and inhibition to maintain a balance between these two driving forces. This mechanism does not seem far-fetched in the light of the multitude of dynamic processes which regulate short and long-term plasticity in neurons both locally at any synapse as well as globally via epigenetic mechanisms.
Activity regularization. We penalize for nonzero neuron output. In this case, each neuron's spike output is considered costly, expressed by the term:

\[ L_{A1} = \sum_i \sum_j \frac{1}{T} \sum_t \phi_{il}(t), \]  

(11)

Applying this regularization term reduces the number of spike events present in the network activity and was inspired by previous works [10]. However, as this activity can only be of positive value, this method by itself has the potential to reduce activity to the point of inducing the dead neuron problem. Therefore, the combination of this approach with potential regularization is expected to both avoid dead neurons and induce temporally sparse activity patterns.

In our AE architecture, this constraint is useful to be applied to a single layer. For applying the activity regularization to the most compressed latent spikes representation (e.g., layer \( l = 3 \)) we define \( L_{A1_{l=3}} \).

Regularized SAE general objective function. Using all regularization terms the SAE takes the following training objective.

\[ L_{SAE} = L_{rec} + \lambda_2 L_{L2} + p_1 L_{P1} + p_2 L_{P2} + a_1 L_{A1} \]  

(12)

Where each regularization term has an associated weight hyper-parameter. Depending on the regularization terms used, the SAE model will present different activation behaviors—See Table 1. A SAE trained with \( L_{P1} \), \( L_{P2} \) and \( L_{A1} \) will force sparsity in the latent representation (hereafter referred to as SAE-sparse). A SAE trained with \( L_{P2} \) and \( L_{A1_{l=3}} \) will provide dense activation without bursting or dying neurons (hereafter referred to as SAE-dense).

For consistency, we define the loss for the baselines using the same notation. The deterministic AE's training objective is: \( L_{AE} = L_{rec} + \lambda_2 L_{L2} \). The VAE’s loss function takes the following form: \( L_{VAE} = L_{rec} + \beta L_{KL} + \lambda_2 L_{L2} \), where \( L_{KL} \) is the Kullback-Leibler divergence term [19].

3 Experiments

We compared three versions of the proposed SAE with a deterministic AE and a VAE—Summarized in Table 1. Both baseline models use the rectified linear unit (ReLU) activation function in all layers, with the exception of the layer constructing the latent representation in the VAE. In contrast to the baseline, all SAE models use a LIF mechanic as a nonlinear activation function. Unlike SAE, the baselines have a learned bias parameter for each neuron. All models were trained on the MNIST dataset for 10 epochs with 5 repetitions. Training and architecture implementation details, hyperparameters analysis and further studies, such as bottleneck impact and latent representation clustering, can be found in the Appendix.

Table 1: Overview of compared models and their respective regularization term weights as well as the reconstruction error and ratio of average intra-class to inter-class latent representation distances.

| Model name | Non-zero regularization weights | \( L_{rec} \) (mean \( \pm \) std) | \( \sigma_{intra}/\sigma_{inter} \) |
|------------|---------------------------------|-----------------------------------|-----------------------------------|
| SAE        | no regularization               | 38.09 \( \pm \) 8.98             | 0.65 \( \pm \) 0.07               |
| SAE-sparse | \( p_1 = 0.005; p_2 = 0.005; a_1 = 0.01 \) | 11.58 \( \pm \) 0.89             | 0.74 \( \pm \) 0.01               |
| SAE-dense  | \( p_2 = 0.01; a_{1_{l=3}} = 0.1 \) | 6.75 \( \pm \) 0.26             | 0.78 \( \pm \) 0.00               |
| AE         | no regularization               | 7.89 \( \pm \) 4.83             | 0.65 \( \pm \) 0.08               |
| AE_{l2}    | \( l_2 = 0.0001 \)             | 4.67 \( \pm \) 0.98             | 0.69 \( \pm \) 0.04               |
| VAE        | \( l_2 = 0.01; \beta = 1.0 \)  | 19.09 \( \pm \) 0.66             | 0.97 \( \pm \) 0.00               |
| \( \beta \)VAE | \( l_2 = 0.01; \beta = 0.1 \) | 5.94 \( \pm \) 0.21             | 0.94 \( \pm \) 0.00               |

All models used in this work use the same architecture depicted in Figure 1. It consists of 6 layers in total, 3 for the encoder and 3 for the decoder. The input is first passed through two convolutional layers for feature map generation with 16 and 32 5x5 kernels respectively [20]. The final layer of the encoder takes in a flattened representation of these feature maps and is fully connected, resulting in a flat latent representation of the data of size \( n_z \). In the decoding process, this latent representation is passed through another fully connected layer and two deconvolutional layers which mirror the
encoder so the output size matches that of the input. While SAE does not use regularization terms, SAE-sparse uses the potential and the activation regularization in all layers. Finally, the SAE-dense only uses the squared potential penalty term and the activity regularization on layer 3 (deepest layer).

Table 1 summarizes the compared models and their reconstruction performance (lower is better) as well as the intra-inter class distance ratio (lower is better), as a measure of clustering. The best performance was reported for the AE and SAE-dense achieved the best spiking performance. This indicates the strong impact of regularization in SAEs unsupervised learning, similarly to VAEs. However, unlike VAEs, SAEs presented better class representation clustering—See Appendix A.5.

3.1 Qualitative Model Comparison

Figure 2: Qualitative model comparison. A Input, Poisson encoding, and model reconstruction of MNIST images for all models. B Comparison of the models learned receptive fields for input and output layers. C Comparison of latent representation of the MNIST examples from panel A across spiking models. D Normalized firing rate distributions across layers of spiking models and the Poisson encoding (top-left in green). Examples of all panels were generated from the model with the respective lowest reconstruction error across the 5 repetitions.

We evaluated the reconstruction quality, the learned filters and the SAE latent representation. Figure 2A shows 10 example MNIST images, one per class, as well as their Poisson encoding and reconstruction by the spiking models and the regularized baseline models. The Poisson encoding (summed and normalized over 100 time steps) closely resembles the original images. The unregularized SAE shows poor reconstruction quality which is visibly improved by regularization methods. The SAE-dense reconstruction quality is comparable to the non-spiking baseline models. The example image for digit “3” constitutes an edge-case as slight differences between the original image and reconstruction of the dense SAE and $\beta$VAE result in output that closely resembles an “8”. In any case, these results are a clear improvement upon earlier approaches to unsupervised learning with SAEs [8, 32] (see also Appendix A.6).

Each models’ 16 convolutional kernels of the input and output layer respectively are expected to learn receptive fields close to Gabor filters to detect lines and edges of different orientations across the MNIST images. Figure 2B shows that all model types still show little structure in their input filters after 10 training epochs after visual inspection. The output layer filters, however, seem to have learned more structure of the underlying data compared to the input filters. This is especially true for the regularized SAE models with clearly identifiable receptive fields. A possible explanation for the difference between input and output filter clarity may be the stronger learning signal at the end of
the network architecture due to vanishing gradients. Conversely, regularized non-spiking baseline models presented poorly recognizable filter structures despite their high image reconstruction quality. 

Figure 2C shows the spiking latent representation (layer 3) of the images. The unregularized SAE is a perfect example of bursting and dying neurons. It consistently shows strong bursting behavior in some neurons and inactivity in others across all examples. However, the representation of the SAE-sparse shows a great reduction of spike events per example compared to the unregularized model with no bursting behavior, but still some neurons are consistently inactive. The SAE-dense shows that these descriptions are partially deceiving. In fact, a large number of neurons in these three neurons’ average firing rate for 100 examples (see Appendix Figure 6 top row center), however, shows that the network architecture due to vanishing gradients. Conversely, regularized non-spiking baseline models presented poorly recognizable filter structures despite their high image reconstruction quality. 

3.2 Quantitative Model Comparison

Regularized SAE Models Successfully Learn MNIST Reconstruction Task Regularized SAE models successfully learn the MNIST reconstruction task and achieve comparable performance to the baseline models, even under different information bottleneck conditions such as noise and reduced latent size – see Figure 3A & C. In particular, the SAE dense performs best among the spiking models. Training the unregularized SAE for MNIST reconstruction for 10 epochs results in high reconstruction error on the validation set (MSE = 38.09 ± 8.98), a large proportion of dead layer three neurons (active neuron ratio, ANR = 0.17 ± 0.20) and strong bursting (constant firing) in other neurons (avg. firing rate of active neurons, AFR = 0.75 ± 0.11). As predicted, the SAE-sparse successfully overcomes the dying neuron problem (ANR = 0.93 ± 0.04; p < 0.01) as well as significantly reduces neuronal bursting (AFR = 0.06 ± 0.02), while improving the reconstruction error (MSE = 11.58 ± 0.89; p < 0.01). The ratio of average number of active neurons per example over all active neurons (RAE) (RAE = 0.61 ± 0.02) suggests a significant increase in spatial sparseness of the latent representation compared to the unregularized model (RAE = 0.95 ± 0.04; p < 0.0001). Visual inspection of layer three neurons average firing rate for 100 examples (see Appendix Figure 6 top row center), however, shows that these descriptions are partially deceiving. In fact, a large number of neurons in these models rarely ever emit spike events across inputs, which lets these neurons appear not dead, but their contribution to information representation is questionable. The SAE-dense also shows no dead neurons (ANR = 1.0 ± 0.0) but, in contrast to the SAE-sparse, a much more balanced activity across active neurons (see Appendix Figure 6 top row right). While SAE-dense shows no spatial sparsity (RAE = 1.0 ± 0.0), the average firing rate of active neurons (AFR = 0.21 ± 0.02; p < 0.001) is still significantly reduced compared to the unregularized model and image reconstruction error (MSE = 6.75 ± 0.26) is significantly lower compared to the other two SAE models (p < 0.001).

The unregularized deterministic AE suffers from the dying neuron problem in the latent representation (ANR = 0.22 ± 0.09) which is comparable to that of the SAE, but with much lower reconstruction error (MSE = 7.89 ± 4.83). AE_{β}, with The weight decay regularization reduces both dead neurons (ANR = 0.41 ± 0.13; p < 0.05) and reconstruction error (MSE = 4.67 ± 0.98) compared to the unregularized AE. While the probabilistic VAE shows no dead neurons in the latent representation due to the lack of the ReLU activation function, the reduction of the KL-divergence loss term weight \( β \) from 1.0 to 0.1 significantly improves reconstruction quality (VAE MSE = 19.09 ± 0.66; \( β \)VAE
Figure 3: Quantitative model comparison under information bottleneck conditions. Reported data was obtained on the validation set after 10 epochs of training with 5 randomly initialized repetitions (seed). A Reducing the size of the latent representation generally increases model error, except for VAE. Both AE models fail to reliably solve the reconstruction task with \( n_z \leq 50 \). B Decreasing the number of simulated time steps in SAE models generally increases reconstruction error. In contrast to the dense SAE, the sparse model cannot solve the task with \( T = 10 \). C Increasing the noise in the input signal generally increases model error with no tested model capable of solving the reconstruction task with \( \epsilon = 0.9 \). D Dense SAE models reliably show a trade-off between number of avg. spikes per example and reconstruction error when simulated step number is varied. E The regularized SAE models reliably learn the task with fewer training examples compared to baseline. F and G SAE batch processing time scales with the number of simulated time steps but not model size.

\[
\text{MSE} = 5.94 \pm 0.21; \ p < 0.0001
\]

Although the image reconstruction quality of all regularized models is comparable (Figure 2A) the reconstruction error of the \( AE_{l2} \) is still lower than \( \beta \)VAE.

Data Efficacy and Batch Processing Time The regularized spiking models need fewer training examples to produce recognizable output images compared to baseline (see Figure 3E). While both sparse and dense SAE show good reconstruction quality reliably after a single training epoch, \( AE_{l2} \) and \( \beta \)VAE take between 1 and 7 or 4 and 6 epochs respectively. This group difference in reconstruction error is significant for epochs 1 through 3 (\( p < 0.0001 \) Kruskal–Wallis). With regard to time, however, due to the simulation of and backpropagation through 100 time steps, the spiking models take much longer for processing a single batch compared to baseline (Figure 3G). The processing time scales linearly with the number of simulated time steps \( T \) (Figure 3F), but not with model size (Figure 3G). This difference is important because it constitutes the key limitation to the scalability of this approach. Backpropagation through time requires that the spiking output and membrane potential of all neurons in the network are stored in memory for each simulated time step. This linear growth of memory usage effectively prevents the simulation of longer sequences. How learning on a continuous stream of data can be realized with this method is still challenging. Alternative learning algorithms must overcome this limitation, for example by relying on approximate transient signals indicating neuron activity. An interesting candidate for this task may be e-prop [4].

4 Conclusion

The results of this work show that the described SAE models consisting of multiple fully trainable layers of LIF neurons are successfully performing end-to-end learning via error backpropagation. We showed that regularization of the membrane potential and spiking output of LIF neurons is an
effective method to considerably improve SAE image reconstruction error while avoiding neural bursting behavior as well as dying neurons. Regularized SAE models successfully learn the MNIST reconstruction task and specifically, the SAE-dense shows reconstruction quality comparable to non-spiking baseline models while requiring less training data but longer processing time. These results constitute an improvement upon earlier approaches to unsupervised learning in SNNs [8,32].

Furthermore, hierarchical clustering of the latent representations reveals similarities between examples of the same stimulus category in regularized SAE models despite the absence of label information during training. This representation structure quality is not present in the probabilistic VAE models, a difference to be kept in mind when using one or the other as a model for early perception. Finally, layer three neuron firing rates are correlated across examples and organized in functional subgroups with no strict borders suggesting a distributed population code with weak redundancies.

Our proposed regularized spiking-autoencoder allows us to control the structure and density of the spiking latent representations. This study is essential for achieving optimal performance in neuromorphic implementations of unsupervised learning models.

References

[1] Subutai Ahmad and Luiz Scheinkman. How can we be so dense? the benefits of using highly sparse representations. arXiv preprint arXiv:1903.11257, 2019.
[2] David Attwell and Simon B. Laughlin. An Energy Budget for Signaling in the Grey Matter of the Brain. Journal of Cerebral Blood Flow & Metabolism, 21(10):1133–1145, oct 2001.
[3] Henri Bal, Dick Epema, Cees de Laat, Rob van Nieuwpoort, John Romein, Frank Seinstra, Cees Snoek, and Harry Wijs Hoff. A medium-scale distributed system for computer science research: Infrastructure for the long term. Computer, 49(5):54–63, 2016.
[4] Guillaume Bellec, Franz Scherr, Anand Subramoney, Elias Hajek, Darjan Salaj, Robert Legenstein, and Wolfgang Maass. A solution to the learning dilemma for recurrent networks of spiking neurons. Nature Communications, 11(1):1–15, 2020.
[5] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. arXiv preprint arXiv:1308.3432, 2013.
[6] Lukas Biewald. Experiment tracking with weights and biases, 2020. Software available from wandb.com.
[7] Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John Guttag. What is the state of neural network pruning? In Machine Learning and Systems, 2020.
[8] Kendra S. Burbank. Mirrored STDP Implements Autoencoder Learning in a Network of Spiking Neurons. PLoS Computational Biology, 11(12):e1004566, 2015.
[9] Yanqing Chen. Mechanisms of Winner-Take-All and Group Selection in Neuronal Spiking Networks. Frontiers in Computational Neuroscience, 11:20, apr 2017.
[10] Benjamin Cramer, Sebastian Billaudelle, Simeon Kanya, Aron Leibfried, Andreas Grübl, Vitali Karasenko, Christian Pehle, Korbirian Schreiber, Yannik Stradmann, Johannes Weis, et al. Training spiking multi-layer networks with surrogate gradients on an analog neuromorphic substrate. arXiv preprint arXiv:2006.07239, 2020.
[11] Mike Davies, Andreas Wild, Garrick Orchard, Yulia Sandamirskaya, Gabriel A Fonseca Guerra, Prasad Joshi, Philipp Plank, and Sumedh R Risbud. Advancing neuromorphic computing with lohi: A survey of results and outlook. Proceedings of the IEEE, 2021.
[12] Trevor Gale, Erich Elsen, and Sara Hooker. The state of sparsity in deep neural networks. arXiv preprint arXiv:1902.09574, 2019.
[13] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. arXiv preprint arXiv:2010.02193, 2020.
[14] Kuan Han, Haiguang Wen, Junxing Shi, Kun-Han Lu, Yizhen Zhang, Di Fu, and Zhongming Liu. Variational autoencoder: An unsupervised model for encoding and decoding fmri activity in visual cortex. NeuroImage, 198:125–136, 2019.
[15] Andreas VM Herz, Tim Gollisch, Christian K Machens, and Dieter Jaeger. Modeling single-neuron dynamics and computations: a balance of detail and abstraction. science, 314(5796):80–85, 2006.
[16] Eric Hunsberger and Chris Eliasmith. Training spiking deep networks for neuromorphic hardware. arXiv preprint arXiv:1611.05141, 2016.

[17] Bernd Illing, Wulfram Gerstner, and Johanni Brea. Biologically plausible deep learning — But how far can we go with shallow networks? Neural Networks, 118:90–101, 2019.

[18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[19] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114, 2013.

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 1097–1105, 2012.

[21] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324, 1998.

[22] Chankyu Lee, Syed Shakib Sarwar, Priyadarshini Panda, Gopalakrishnan Srinivasan, and Kaushik Roy. Enabling Spike-based Backpropagation for Training Deep Neural Network Architectures. Frontiers in Neuroscience, 14, mar 2019.

[23] Jun Haeng Lee, Tobi Delbruck, and Michael Pfeiffer. Training Deep Spiking Neural Networks Using Backpropagation. Frontiers in Neuroscience, 10(NOV):508, nov 2016.

[24] Ifat Levy, Uri Hasson, and Rafael Malach. One Picture Is Worth at Least a Million Neurons. Current Biology, 14(11):996–1001, jun 2004.

[25] Lu Lu, Yeonjong Shin, Yanhui Su, and George Em Karniadakis. Dying relu and initialization: Theory and numerical examples. arXiv preprint arXiv:1903.06733, 2019.

[26] Emile Mathieu, Tom Rainforth, N Siddharth, and Yee Whye Teh. Disentangling disentanglement in variational autoencoders. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, Proceedings of the 36th International Conference on Machine Learning, volume 97 of Proceedings of Machine Learning Research, pages 4402–4412. PMLR, 09–15 Jun 2019.

[27] Cristian Meo and Pablo Lanillos. Multimodal vae active inference controller. arXiv preprint arXiv:2103.04412, 2021.

[28] Emre O. Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate Gradient Learning in Spiking Neural Networks: Bringing the Power of Gradient-based optimization to spiking neural networks. IEEE Signal Processing Magazine, 36(6):51–63, 2019.

[29] Peter O’Connor, Efstratios Gavves, Matthias Reisser, and Max Welling. Temporally efficient deep learning with spikes. 6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings, pages 1–19, 2018.

[30] Peter O’Connor and Max Welling. Deep spiking networks. arXiv preprint arXiv:1602.08323, 2016.

[31] Garrick Orchard, Ajinkya Jayawant, Gregory K Cohen, and Nitish Thakor. Converting static image datasets to spiking neuromorphic datasets using saccades. Frontiers in neuroscience, 9:437, 2015.

[32] Priyadarshini Panda and Kaushik Roy. Unsupervised regenerative learning of hierarchical features in Spiking Deep Networks for object recognition. In Proceedings of the International Joint Conference on Neural Networks, pages 299–306. Institute of Electrical and Electronics Engineers Inc., 2016.

[33] Michael G. Paulin and Andre van Schaik. Bayesian Inference with Spiking Neurons. Encyclopedia of Computational Neuroscience, pages 1–4, jun 2014.

[34] Michael Pfeiffer and Thomas Pfeil. Deep learning with spiking neurons: opportunities and challenges. Frontiers in neuroscience, 12:774, 2018.

[35] Siyuan Qiao, Zhe Lin, Jianming Zhang, and Alan L Yuille. Neural rejuvenation: Improving deep network training by enhancing computational resource utilization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 61–71, 2019.

[36] Rodrigo Quian Quiroga and Gabriel Kreiman. Measuring sparseness in the brain: Comment on Bowers (2009). Psychological Review, 117(1):291–297, 2010.
[37] Ali Razavi, Aaron van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *arXiv preprint arXiv:1906.00446*, 2019.

[38] Deboleena Roy, Priyadarshini Panda, and Kaushik Roy. Synthesizing images from spatio-temporal representations using spike-based backpropagation. *Frontiers in Neuroscience*, 13(JUN), 2019.

[39] Catherine D Schuman, Thomas E Potok, Robert M Patton, J Douglas Birdwell, Mark E Dean, Garrett S Rose, and James S Plank. A survey of neuromorphic computing and neural networks in hardware. *arXiv preprint arXiv:1705.06963*, 2017.

[40] Amirhossein Tavanaei and Anthony S Maida. Bio-inspired spiking convolutional neural network using layer-wise sparse coding and stdp learning. *arXiv preprint arXiv:1611.03000*, 2016.

[41] Davide Zambrano, Roeland Nusselder, H. Steven Scholte, and Sander M. Bohté. Sparse Computation in Adaptive Spiking Neural Networks. *Frontiers in Neuroscience*, 12(JAN):987, jan 2019.

[42] Friedemann Zenke and Surya Ganguli. SuperSpike: Supervised Learning in Multilayer Spiking Neural Networks. *Neural Computation*, 30(6):1514–1541, jun 2018.

[43] Malu Zhang, Hong Qu, Ammar Belatreche, Yi Chen, and Zhang Yi. A Highly Effective and Robust Membrane Potential-Driven Supervised Learning Method for Spiking Neurons. *IEEE Transactions on Neural Networks and Learning Systems*, 30(1):123–137, 2019.
A Appendix

A.1 Training details

In this work, model weight parameters $\theta_l$ for all layers $l \in L$ are initialized randomly according to:

$$\theta_l \sim N\left(0, \sqrt{\frac{2}{n_l}}\right),$$

where $n_l$ is the input size in the fully connected case and kernel width $\times$ kernel height $\times$ number of input channels for convolutional and deconvolutional layers.

After initialization, model parameters are updated iteratively through backpropagation on batches of $N = 64$ training examples each for 10 epochs with a learning rate of $\alpha = 0.0005$. The exact step size of each parameter value $\Delta \theta$ is determined by the ADAM optimization algorithm at standard settings [18].

A.2 Networks & Learning Parameters

The key parameters describing the models and learning process are summarized in Table 3 and Table 1. All parameter values were found in preliminary experiments through empirical testing. The reported default values are used throughout the reported experiments unless explicitly specified. The number of trainable parameters $\theta$ varies across models and scales linearly with layer 3 latent representation size $n_z$ according to Table 2.

Table 2: Trainable model parameters depend on the size of the latent representation. SAE models have fewer parameters compared to the baseline models due to the lack of bias parameters. VAE model parameters are increased because the number of neurons in layer three are effectively doubled by predicting both $\mu$ and $\log \sigma^2$, each with $n_z$ individual neurons.

| latent size $n_z$ | SAE | AE | VAE |
|-------------------|-----|----|-----|
| 10                | 282,400 | 295,210 | 423,220 |
| 20                | 538,400 | 551,220 | 807,240 |
| 50                | 1,306,400 | 1,319,250 | 1,959,300 |
| 100               | 2,586,400 | 2,599,300 | 3,879,400 |

Table 3: Overview of relevant hyper-parameters and their default values. The last three parameters are only relevant in the context of the SAE.

| Parameter | Default value |
|-----------|---------------|
| Architecture | Kernel size 5 x 5 |
|             | Convolution channels 16, 32 |
|             | Latent size $n_z$ 100 |
| Training   | Epochs 10 |
|            | Batch size $N$ 64 |
|            | Learning rate $\alpha$ 0.0005 |
| Encoding   | Noise $\epsilon$ 0.0 |
|            | Scaling factor $s$ 0.2 |
| Temporal dynamics | Time steps $T$ 100 |
|            | Potential decay $\tau$ 0.99 |

Autoencoder Parameters First, the baseline AE model was used to determine a suitable learning rate and L2 regularization parameter based on the lowest reconstruction error. The results of this search are summarized in Table 4. The results show a large effect of the learning rate on the results with best and most reliable performance at $\alpha = 0.0005$. To enable fair model comparison, this learning rate was also chosen for all other models in all following experiments. Learning rates of $\alpha \geq 0.01$ resulted in the AE models to return all-zero image reconstructions, equivalent to a $L_{rec}$
of approximately 89. Further, these results show that in all reported cases the AE models have at least 50% dead neurons in layer 3, limiting each models information representation capacity in the latent space. The L2 regularization has a slight improvement on AE reconstruction error, with an optimum for \( l_2 = 0.0001 \). However, L2 regularization was not suitable for addressing the dying neuron problem.

Table 4: AE parameter search. First, a suitable learning rate was determined as \( \alpha = 0.0005 \) based on the lowest reconstruction loss without regularization. Later, L2 loss weight parameter was determined as \( l_2 = 0.0001 \). All runs were done over 10 epochs on the full dataset. Inactive neurons are reported for layer 3 only. Parameters chosen for further experiments are highlighted in bold. INP = inactive neuron proportion of layer 3.

| Parameter | Value | \( L_{rec} \) | INP |
|-----------|-------|---------------|-----|
| \( \alpha \) | 0.000001 | 63.78 ± 5.07 | 0.66 ± 0.07 |
| | 0.00001 | 30.14 ± 12.89 | 0.67 ± 0.22 |
| **0.0005** | **5.65** ± **0.94** | **0.67** ± **0.08** |
| 0.001 | 15.68 ± 9.86 | 0.72 ± 0.15 |
| 0.01 | 45.86 ± 43.61 | 0.65 ± 0.30 |
| 0.1 | 89.46 ± 0.00 | 0.91 ± 0.04 |
| \( l_2 \) | 5.65 ± 0.94 | 0.67 ± 0.08 |
| 0 | 5.65 ± 1.41 | 0.66 ± 0.10 |
| 0.00001 | 4.54 ± 1.49 | 0.56 ± 0.10 |
| **0.0001** | **4.50** ± **1.38** | **0.62** ± **0.11** |
| 0.001 | 5.57 ± 1.89 | 0.72 ± 0.10 |
| 0.01 | 6.54 ± 1.18 | 0.78 ± 0.04 |
| 0.1 | 7.98 ± 1.25 | 0.81 ± 0.05 |

**Variational Autoencoder Parameters** As the next step, VAE regularization parameters were determined and the results of this search are summarized in Table 5. In this case, first the weight of the Kullback–Leibler divergence term \( \beta \) was fixed at its standard value of 1 to find a suitable L2 loss weight. In contrast to the AE model, for the VAE sufficient L2 regularization (\( l_2 \geq 0.001 \)) was necessary to obtain any useful image reconstructions. The optimal parameter was found at \( l_2 = 0.01 \) and is therefore 100 times larger compared to the AE.

In contrast, lower values for \( \beta \) show a strong decrease in model reconstruction error. In fact, ignoring the Kullback–Leibler divergence term by setting \( \beta = 0 \) showed the lowest reconstruction error across all experiments. However, in the case of low values for \( \beta \), the \( \sigma \) vector predicted by the encoder showed very small values (data not shown), rendering the "variation" quasi non-existent. This is not surprising, because the key function of this regularization term is to put a soft constraint on the values of \( \sigma \) to be close to 1, so that \( z \) is sampled with variance. The VAE with low values for \( \beta \) can therefore be seen as a quasi-deterministic AE model. The difference in performance of these models (VAE \( \beta = 0 L_{rec} = 1.646 \pm 0.943 \)) compared to the best AE configuration (AE \( l_2 = 0.0001 L_{rec} = 4.495 \pm 1.382 \)) can therefore be explained by the lack of a ReLU activation function on the predicted \( \mu \) vector, and the resulting absence of dead neurons in layer 3. This is a strong indication that the dying neuron problem may have a large effect on model performance and should therefore be avoided as much as possible. Although the image reconstruction error is lowest in cases of low \( \beta \), only the cases of \( \beta = 1 \) and \( \beta = 0.1 \) were considered for further analysis, as in these cases the variation is still functioning with values of \( \sigma \) close to 1.

**Spiking Autoencoder Parameters** For the SAE models, a scaling parameter \( s \) is required for both information encoding and decoding. Specifically, this parameter directly influences the expected firing rate of neurons resulting from the Poisson encoding. A value of \( s = 1 \) refers to a spiking probability of 100% at each time step if the respective pixel value of an example image is 1 (assuming normalized pixel values). Similarly, the output layer would be expected to show the same constant bursting behavior for respective neurons to obtain a low reconstruction error. High firing rates may increase the precision with which the floating point pixel values are translated into sequences of discrete spike events. However, bursting at maximum firing rates is considered inefficient and aimed
Table 5: VAE parameter search. For comparability, the same learning rate as AE was used with $\alpha = 0.0005$. First, the L2 loss weight parameter was determined as $l_2 = 0.01$ with a fixed $\beta = 1.0$. Later, the effect of different values for $\beta$ were tried with L2 regularization. All runs were done over 10 epochs on the full dataset. Parameters chosen for further experiments are highlighted in bold.

| Parameter | Value | $L_{rec}$ |
|-----------|-------|-----------|
| $l_2$     | 0     | 89.46 ± 0.00 |
|           | 0.000001 | 89.46 ± 0.00 |
|           | 0.00001 | 89.46 ± 0.00 |
|           | 0.001  | 89.46 ± 0.00 |
|           | 0.01   | 38.40 ± 29.50 |
|           | **0.01** | **19.14 ± 0.61** |
|           | 0.1    | 20.76 ± 0.23 |
| $\beta$   | 0     | 1.65 ± 0.94 |
|           | 0.001  | 1.64 ± 1.41 |
|           | 0.01   | 2.29 ± 1.49 |
|           | **0.1** | **5.94 ± 1.38** |
|           | 1      | 19.14 ± 0.61 |
|           | 10     | 52.90 ± 0.08 |

to be avoided with the regularization terms. It therefore seems not helpful to induce and expect bursting behavior, while at the same time punishing it. Preliminary experiments showed that of all tested values for $0.1 \leq s \leq 1$, results were most promising at $s = 0.2$ (data not shown). Note that this does not mean that the expected normalized average firing rate of neurons in any layer should be at 0.2. The MNIST dataset has a mean normalized pixel value of approximately $0.13 ± 0.31$. Therefore, the expected normalized average firing rate of any Poisson encoded images (and ideally layer 6 output) is $0.2 \times 0.13 = 0.026$. This matches well with the recorded average 0.025 (see green histogram in Figure 2D). For the purposes of this work, this is considered an overall sparse encoding.

Beyond the scaling parameter, we were looking for a set of regularization parameter values that that result low reconstruction error $L_{rec}$ as well as a low percentage of inactive layer 3 neurons (INR), preferably 0. Preconceptions are different about the latent representation spike density, that is the average normalized firing rate of active neurons (AFR) only. Here, no exact optimum is desired. Instead spike density should neither be too high, indicating inefficient bursting behavior, nor should it be 0, indicating no activity whatsoever. Monitoring this variable in conjunction with reconstruction error is particularly interesting, as we find a trade-off between the total number of spike events and the precision with which information is encoded in the latent representation. In all cases, standard deviations for these outcome variables should be low, indicating robust effects of the regularization method.

Based on the desired effect of the different regularization types, introduced in subsection 2.3, we expect potential regularization (P1 and P2) to prevent dead neurons (decrease percent inactive). Furthermore, activity regularization (A1) is expected to reduce the spike density, but may also increase the number of dead neurons when penalizing spiking behavior too strong. As this term penalizes every spike independent of the respective neurons’ firing rate, we expect these rates to be generally lower compared to the unregularized case, but not necessarily evenly distributed across neurons. Finally, in order to both prevent dying neurons and bursting behavior, we expect that a combination of potential and activity regularization to be most suitable.

SAE regularization methods were initially tested in preliminary experiments (1 epoch only) mostly independent of one another to find parameter values that can improve model performance. The results of this search are presented in Table 6. The default network without any regularization performs poorly on the image reconstruction task with around 86% of layer three neurons dead, while the remaining neurons engage in high firing rates. Weight regularization slightly improves the reconstruction loss and performance reliability, but instead increase the average spike density. The number of inactive neurons remains largely unchanged. Therefore, weight regularization seems unsuitable to address the problems of dead and bursting neurons in the SAE network.
In contrast, potential regularization and activity regularization reduce reconstruction loss, spike density, and the number of inactive neurons for various parameter values. Interestingly, in all cases a parameter value of 0.01 for $p_1$, $p_2$, and $a_1$ shows the lowest reconstruction error. Stronger $P_1$ and $A_1$ regularization results in the network not learning the task, and weaker regularization results are comparable to the unregularized network. Although some parameter values are clearly preferable to others, in general potential regularization, specifically $P_2$, seems suitable to avoid dead neurons as expected.

Table 6: In this first phase of parameter search for the SAE, regularization methods were tested primarily independent of one another. Average firing rate of active neurons (AFR) and inactive neuron proportion (INP) are reported for layer 3 only. For these experiments, networks were trained for a single epoch to shorten overall runtime. Colors highlight high values in red and low values in green per column for easier visual comparison.

| Parameter | Value | $L_{rec}$ | AFR | INP |
|-----------|-------|-----------|-----|-----|
| None      |       | 38.66 ± 10.19 | 0.51 ± 0.15 | 0.86 ± 0.18 |
| $l_2$     | 0.1   | 37.74 ± 6.07 | 0.64 ± 0.10 | 0.85 ± 0.14 |
|           | 0.01  | 36.24 ± 6.39 | 0.59 ± 0.09 | 0.87 ± 0.11 |
|           | 0.001 | 35.36 ± 7.80 | 0.56 ± 0.12 | 0.85 ± 0.14 |
|           | 0.0001| 33.57 ± 5.56 | 0.54 ± 0.11 | 0.86 ± 0.12 |
|           | 0.00001| 33.44 ± 4.72 | 0.53 ± 0.09 | 0.84 ± 0.13 |
| $p_1$     | 0.1   | 89.39 ± 0.00 | 0.00 ± 0.00 | 1.00 ± 0.00 |
|           | 0.01  | 25.19 ± 1.23 | 0.02 ± 0.01 | 0.07 ± 0.06 |
|           | 0.001 | 27.95 ± 5.97 | 0.13 ± 0.03 | 0.66 ± 0.04 |
|           | 0.0001| 33.89 ± 5.64 | 0.51 ± 0.38 | 0.80 ± 0.14 |
|           | 0.00001| 30.51 ± 2.81 | 0.67 ± 0.59 | 0.77 ± 0.16 |
| $p_2$     | 0.1   | 22.71 ± 1.23 | 0.04 ± 0.00 | 0.00 ± 0.00 |
|           | 0.01  | 18.55 ± 3.21 | 0.07 ± 0.02 | 0.00 ± 0.00 |
|           | 0.005 | 19.61 ± 3.35 | 0.11 ± 0.02 | 0.23 ± 0.07 |
|           | 0.001 | 23.86 ± 5.18 | 0.08 ± 0.02 | 0.23 ± 0.08 |
|           | 0.0001| 28.82 ± 4.91 | 0.24 ± 0.11 | 0.54 ± 0.16 |
|           | 0.00001| 31.65 ± 10.07| 0.45 ± 0.12 | 0.80 ± 0.13 |
| $a_1$     | 0.1   | 56.21 ± 4.57 | 0.00 ± 0.00 | 0.00 ± 0.00 |
|           | 0.01  | 22.29 ± 2.95 | 0.05 ± 0.03 | 0.17 ± 0.16 |
|           | 0.001 | 30.28 ± 5.81 | 0.44 ± 0.09 | 0.89 ± 0.04 |
|           | 0.0001| 35.13 ± 7.45 | 0.54 ± 0.07 | 0.85 ± 0.13 |
|           | 0.00001| 33.77 ± 4.58 | 0.50 ± 0.05 | 0.86 ± 0.12 |

After these promising initial results for potential and activity regularization methods in preliminary experiments, several combinations of loss terms were evaluated in longer experiments (10 epochs). The results are summarized in Table 7. Interestingly, the unregularized model shows about the same reconstruction loss and number of dead neurons after 10 epochs of training as after a single epoch, but the average spike density of surviving neurons increased from 0.51 ± 0.15 to 0.75 ± 0.11. In contrast, all tested combinations of potential and activity regularization result in the SAE network learning the image reconstruction task successfully. Further, dead neurons can be reduced reliably and avoided all together in many cases.

Table 7: The unregularized model shows about the same reconstruction loss and number of dead neurons after 10 epochs of training as after a single epoch, but the average spike density of surviving neurons increased from 0.51 ± 0.15 to 0.75 ± 0.11. In contrast, all tested combinations of potential and activity regularization result in the SAE network learning the image reconstruction task successfully. Further, dead neurons can be reduced reliably and avoided all together in many cases.

| Parameter | Value | $L_{rec}$ | AFR | INP |
|-----------|-------|-----------|-----|-----|
| None      |       | 38.66 ± 10.19 | 0.51 ± 0.15 | 0.86 ± 0.18 |
| $l_2$     | 0.1   | 37.74 ± 6.07 | 0.64 ± 0.10 | 0.85 ± 0.14 |
|           | 0.01  | 36.24 ± 6.39 | 0.59 ± 0.09 | 0.87 ± 0.11 |
|           | 0.001 | 35.36 ± 7.80 | 0.56 ± 0.12 | 0.85 ± 0.14 |
|           | 0.0001| 33.57 ± 5.56 | 0.54 ± 0.11 | 0.86 ± 0.12 |
|           | 0.00001| 33.44 ± 4.72 | 0.53 ± 0.09 | 0.84 ± 0.13 |
| $p_1$     | 0.1   | 89.39 ± 0.00 | 0.00 ± 0.00 | 1.00 ± 0.00 |
|           | 0.01  | 25.19 ± 1.23 | 0.02 ± 0.01 | 0.07 ± 0.06 |
|           | 0.001 | 27.95 ± 5.97 | 0.13 ± 0.03 | 0.66 ± 0.04 |
|           | 0.0001| 33.89 ± 5.64 | 0.51 ± 0.38 | 0.80 ± 0.14 |
|           | 0.00001| 30.51 ± 2.81 | 0.67 ± 0.59 | 0.77 ± 0.16 |
| $p_2$     | 0.1   | 22.71 ± 1.23 | 0.04 ± 0.00 | 0.00 ± 0.00 |
|           | 0.01  | 18.55 ± 3.21 | 0.07 ± 0.02 | 0.00 ± 0.00 |
|           | 0.005 | 19.61 ± 3.35 | 0.11 ± 0.02 | 0.23 ± 0.07 |
|           | 0.001 | 23.86 ± 5.18 | 0.08 ± 0.02 | 0.23 ± 0.08 |
|           | 0.0001| 28.82 ± 4.91 | 0.24 ± 0.11 | 0.54 ± 0.16 |
|           | 0.00001| 31.65 ± 10.07| 0.45 ± 0.12 | 0.80 ± 0.13 |
| $a_1$     | 0.1   | 56.21 ± 4.57 | 0.00 ± 0.00 | 0.00 ± 0.00 |
|           | 0.01  | 22.29 ± 2.95 | 0.05 ± 0.03 | 0.17 ± 0.16 |
|           | 0.001 | 30.28 ± 5.81 | 0.44 ± 0.09 | 0.89 ± 0.04 |
|           | 0.0001| 35.13 ± 7.45 | 0.54 ± 0.07 | 0.85 ± 0.13 |
|           | 0.00001| 33.77 ± 4.58 | 0.50 ± 0.05 | 0.86 ± 0.12 |

As expected, a trade-off between spike density and reconstruction error is clearly visible in Table 7. Generally speaking, models that show the lowest reconstruction errors (below 7) have a higher spike density (around 0.2) than models with slightly higher reconstruction error (between 10 and 12). This trade-off is found again in later experiments (see Figure 3D & Figure 4C). Interestingly, layer 3 spike density is reduced more effectively by assigning this cost to spikes emitted in all layer, while $A_1(l=3)$ paired with $P_2$ results in the overall lowest reconstruction error for compared spiking models.

Based on these findings, three SAE models are selected for further analysis. The first model is the unregularized SAE as an additional baseline. Second choice is a model configuration for sparse latent
representations based on potential and A1 regularization, hereafter referred to as \textit{sparse SAE}. Finally, a model from the other end of the error-sparsity trade-off is selected with potential and A1(l=3) regularization, hereafter referred to as \textit{dense SAE}. The parameter values for all three selected SAE models are summarized in Table 1.

Table 7: In this second phase of parameter search for the SAE, regularization methods were tested primarily combined of one another. Average firing rate of active neurons (AFR) and inactive neuron proportion (INP) are reported for layer 3 only. In these tests L1 regularization is excluded. All runs were done over 10 epochs on the full dataset. Parameters chosen for further experiments are highlighted in bold. Colors highlight high values in red and low values in green per column for easier visual comparison.

| $l_2$  | $p_1$  | $p_2$  | $a_1$  | $a_{1,l=3}$ | $L_{rec}$  | AFR    | INP    |
|--------|--------|--------|--------|-------------|------------|--------|--------|
| 0.01   |        |        |        |             | 38.09 ± 8.96 | 0.75 ± 0.11 | 0.83 ± 0.20 |
| 0.001  |        |        |        |             | 37.03 ± 8.85 | 0.69 ± 0.11 | 0.86 ± 0.18 |
| 0.0001 |        |        |        |             | 39.63 ± 10.14 | 0.77 ± 0.16 | 0.85 ± 0.20 |
| 0.00001|        |        |        |             | 38.89 ± 10.71 | 0.75 ± 0.17 | 0.86 ± 0.17 |
|        |        |        |        |             | 36.29 ± 6.78 | 0.69 ± 0.14 | 0.84 ± 0.17 |
| 0.01   | 0.005  | 0.005  | 0.01   |             | 12.92 ± 4.76 | 0.09 ± 0.07 | 0.06 ± 0.06 |
| 0.005  | 0.005  | 0.005  | 0.01   |             | 8.27 ± 1.33  | 0.15 ± 0.04 | 0.00 ± 0.00 |
| 0.005  | 0.005  | 0.005  | 0.01   |             | 11.58 ± 0.89 | 0.06 ± 0.02 | 0.07 ± 0.04 |
| 0.01   | 0.01   |        |        |             | 7.85 ± 1.92  | 0.22 ± 0.01 | 0.00 ± 0.00 |
| 0.01   | 0.01   |        |        |             | 9.08 ± 0.44  | 0.11 ± 0.01 | 0.02 ± 0.01 |
| 0.01   | 10     |        |        |             | 15.07 ± 1.39 | 0.04 ± 0.00 | 0.15 ± 0.04 |
| 0.01   | 1      |        |        |             | 8.43 ± 0.49  | 0.11 ± 0.01 | 0.00 ± 0.00 |
| 0.01   | 0.1    |        |        |             | 6.94 ± 0.18  | 0.20 ± 0.01 | 0.00 ± 0.00 |
| 0.01   | 0.01   |        |        |             | 6.75 ± 0.26  | 0.21 ± 0.02 | 0.00 ± 0.00 |

A.3 Software & Hardware Implementation

All code used for this project was written in Python 3.7 and is available on \url{https://github.com/jhuebotter/SpikingVAE}. For deep learning the pytorch framework (version 1.5) was extended with several custom classes to accommodate the LIF functionality. Result images are generated with matplotlib (version 3.1.3) and seaborn (version 0.11) and statistical tests were conducted using scipy (version 1.5.2). Weights & Biases were used for experiment tracking \cite{6}.

The training, validation, and testing of all models was performed on RTX 2080Ti GPUs (Nvidia, USA) as part of the Distributed ASCI Supercomputer 5 (DAS-5) server \cite{3}.

A.4 MNIST Dataset

The dataset chosen is the Modified National Institute of Standards and Technology (MNIST) database \cite{21}. The dataset contains 28 x 28 gray-scale images and labels of handwritten digits from 0 to 9 with 60.000 examples for training and 10.000 for testing. All images are scaled in the range [0, 1] by default and consist only of a single channel. MNIST has been utilized as a well studied benchmark in computer vision tasks for the past two decades, and therefore is an easy first choice for investigating performance of novel neural networks against established methods.

Typically, SNNs receive continuous streams of time-varying stimuli, which MNIST images in their default representation are not. Although the images can be and are encoded in a way that includes the dimension of time as described in Equation 6, the underlying source of the stimuli is still static. Arguably, SNNs may not be capable of utilizing their full potential in such use cases when compared to their non-spiking counterparts. Alternative approaches do exist, such as Neuromorphic-MNIST by \cite{31}, who converted the original dataset examples into a temporal series of spikes using event-based...
cameras and saccade-like movements across the input space. This dataset was not used in this work, however, to maintain comparability between the novel SAE and baseline models in the input reconstruction task and remains a possible extension for further research.

A.4.1 Information Bottlenecks

For the following experiments the models are trained and tested under different information bottleneck conditions. Each time the value of a single variable is changed from its default value (see Table 3). A visual summary of key findings is described in Figure 3 and Figure 4.

Reducing the size of the latent representation \( n_z \) generally increases model error, except for VAE (Figure 3A). Both AE models fail to reliably solve the reconstruction task with \( n_z \leq 50 \). The degree by which the SAE model performance seems affected by the reduced latent size seems to lie between the AE model (drastic change in model performance) and VAE (small change in model performance).

As expected, increasing the noise \( \epsilon \) in the input signal generally increases reconstruction error across models (Figure 3C). No tested model was capable of solving the reconstruction task with \( \epsilon = 0.9 \). However, compared to the AE models, SAE models show lower variance in reconstruction error with \( \epsilon = 0.6 \).

As expected, decreasing the number of simulated time steps \( T \) in SAE models generally increases reconstruction error (Figure 3B). In contrast to the dense SAE, the sparse model cannot solve the task with \( T = 10 \). Dense SAE models reliably show a trade-off between the number of avg. spikes per example and the reconstruction error when simulated step number is varied (Figure 3D). While a linear regression shows a significant negative correlation between both variables (\( r = -0.83, p < 1E^{-5} \)), visual inspection shows that this relationship is likely not linear, but rather exponential.

Changes in the membrane potential decay \( \tau \) effects SAE models’ reconstruction performance differently (Figure 4A). While the sparse SAE shows optimal performance for \( \tau = 0.9 \), the dense model performs worst in this case and best with \( \tau = 0.99 \). A possible explanation for the latter could be that the optimization of other model parameters was conducted with the default decay of \( \tau = 0.99 \). This, however, does not explain why the trend of model performance is reversed from a sparse to a dense model. As expected, increasing the decay variable generally leads to a reduction in avg. spikes per example across spiking models (Figure 3B), with some exceptions for the dense model. Interestingly, sparse SAE model error shows significant negative correlation with reconstruction error (\( r = -0.83, p < 1E^{-5} \)) with visible influence of random initialization across repetitions (Figure 4C).

In summary, model behavior is affected by information bottlenecks mostly as expected. The regularized SAE models are somewhat robust to added noise in the input, which was as also reported for SNNs applied to MNIST classification [43, 40]. However, they show no clear advantage in this domain compared to non-spiking baseline models. In general, the spiking models’ robustness towards changes in noise and latent size seems to lie above the AE models, but below VAE models.

Figure 4: A Changes in the membrane potential decay \( \tau \) effects SAE models’ reconstruction performance differently. B Increasing the decay variable generally leads to a reduction in avg. spikes per example across spiking models. C Sparse SAE model error shows significant negative correlation with reconstruction error with visible differences across randomly initialized repetitions.
A.5  Latent Representation Analysis

In this section, the models’ latent representations of MNIST examples are inspected further. For this purpose, of all fully trained models with default parameters, the instances lowest image reconstruction error are chosen for processing a single batch of 100 examples from the validation set with 10 examples per class.

All three compared models encode the observed data in a compressed latent representation $z$, but the nature of this representation is model-specific. For the AE model, the latent representation $z$ of $x$ is a vector of $n_z$ positive rational elements $z_i \in \{Q \geq 0\}$. In the VAE, $z$ is sampled from the multivariate distribution parameterized by the encoder output $\mu$ and $\log(\sigma^2)$ as $z \sim N(\mu(x'), \sigma(x'))$. Therefore, for the VAE model, the latent representation $z$ of $x$ is a vector of $n_z$ rational elements $z_i \in Q$. Finally, the SAE has a temporal dimension in its representation of $z$, so that it has the shape $n_z \times T$, but every element is either 0 or 1. As the SAE encoding and decoding methods are based on rate code, the latent representation of this model is summarized as the average firing rates per neuron in the following analysis to enable comparison with the baseline models. Latent representations are scaled in the range $[0, 1]$ for AE and SAE models and in the range $[-1, 1]$ for VAE models.

A.5.1  Latent Representation Clustering

For the first comparison, the normalized layer 3 activity is represented in a matrix of 100 examples x 100 neurons for each model, a selection of which is shown in Figure 5. Each column representing a single MNIST example is marked with a color representing that example’s class label. Finally, an out-of-the-box hierarchical clustering algorithm is used to calculate linkage between both individual rows and columns based on their Euclidean distance and sort the matrix according to the resulting dendogram. Alternative distance metrics are evaluated in Table 8. The resulting color band on top of each matrix can thereby give an indication, if examples of the same class are grouped together based on similar activity pattern in the latent representation of each model, without any model ever having seen the label data.

The activity matrix of the unregularized SAE model (Figure 5 top left) clearly shows that approximately half of its layer 3 neurons are inactive across all examples, while the other half is engaged in bursting behavior most of the time. The sparse SAE model’s activity matrix (Figure 5 top center) as expected, shows a drastically reduced average firing rate across neurons. It also shows that while technically there are no dead neurons, the majority of neurons actually exhibit very low firing rates consistently across examples. This clearly demonstrates that the regularization method prevents dead neurons and bursting behavior, but it does not at all guarantee that all neurons contribute to the representation of information equally with comparable firing rates. In contrast, the dense SAE model’s activity matrix (Figure 5 top right) shows that bursting and dead neurons are prevented successfully. In this case, regularization enables a more balanced distribution of activity across neurons compared to both other spiking models with an average firing rate comparable to that of the more active group of neurons of the sparse SAE. The AE$_{l2}$ model’s matrix (Figure 5 bottom left) shows a substantial amount of dead neurons. This suggests that L2 regularization is not a suitable method to prevent the dying ReLU problem, which makes sense. In contrast, the VAE (Figure 5 bottom center) exhibits no dead neurons and very balanced yet independent (non-similar) patterns of neuronal activity across examples. This behavior is expected because this model’s layer 3 has no ReLU activation function and the KL-divergence regularization term is an attempt to have neuron activity distributed $\sim N(0, 1)$.

The clustering of examples shows, for the unregularized SAE, most examples of digit 1 are next to each other and some smaller clusters of 7s and 0s exist, but generally the ability of the clustering algorithm to find semantically meaningful structure in the activity matrix is limited. Interestingly, the color bar of both regularized SAE models show more structure in the latent representation, with large groups of the same color clustered together. A similar structure can be found in the AE$_{l2}$ model. For the VAE, however, hierarchical clustering is not able to identify any structure in the latent representation that suggests the class of each can be identified based on similar neuron activity patterns.

19
Table 8: The average distances between latent representations of MNIST examples within the same class $\tilde{s}_{\text{intra}}$ are expected to be smaller than to examples of a different stimulus category $\tilde{s}_{\text{inter}}$, if structural similarity is maintained in the learned representation. The ratios of $\tilde{s}_{\text{intra}}/\tilde{s}_{\text{inter}}$ suggest that this is the case for SAE and AE models, but not for VAE across several distance metrics.

|       | euclidean | stand. euclidean | squared euclidean | manhattan | correlation |
|-------|-----------|------------------|-------------------|-----------|-------------|
| SAE   | 0.65 ± 0.07| 0.69 ± 0.10      | 0.51 ± 0.06       | 0.62 ± 0.06| 0.52 ± 0.05|
| SAE sparse | 0.74 ± 0.01| 0.86 ± 0.02      | 0.59 ± 0.01       | 0.75 ± 0.01| 0.58 ± 0.01|
| SAE dense | 0.78 ± 0.00| 0.80 ± 0.00      | 0.64 ± 0.00       | 0.77 ± 0.00| 0.66 ± 0.01|
| AE    | 0.65 ± 0.08| 0.70 ± 0.07      | 0.46 ± 0.10       | 0.66 ± 0.08| 0.46 ± 0.11|
| AE$_{\ell^2}$ | 0.69 ± 0.04| 0.80 ± 0.03      | 0.52 ± 0.05       | 0.73 ± 0.03| 0.53 ± 0.05|
| VAE   | 0.97 ± 0.00| 0.98 ± 0.00      | 0.95 ± 0.00       | 0.97 ± 0.00| 0.95 ± 0.00|
| $\beta$VAE | 0.94 ± 0.00| 0.95 ± 0.00      | 0.88 ± 0.00       | 0.93 ± 0.00| 0.88 ± 0.00|

Figure 5: Comparison of the models’ latent representation based on normalized activity. Regularization of SAE models prevents dead neurons as well as bursting behavior and activity patterns can successfully be grouped by example labels using hierarchical clustering.

A.5.2 Example Activity Correlation

In this section, the column values (examples) of the normalized activity matrices from Figure 5 are correlated with one another for each model. This allows identification of clusters of examples with similar neuronal representation.

The SAE example cluster map (Figure 6 top left) shows that many example representation pairs have a strong positive correlation and a moderate negative correlation in between images of the classes 0 and 1. Regularization of the spiking model results in a more uniform positive correlation between examples with no negative correlations (Figure 6 top center and top right). The dense SAE, however, has a clearly reduced average correlation compared to both other spiking models. This is another indication that indeed activity is more balanced across neurons. The correlation matrix of the AE$_{\ell^2}$ models (Figure 6 bottom left) strongly resembles that of the sparse SAE. This similarity in high overall example correlations is probably due to the clear trend for some neurons to be consistently more or less active across examples in both models. As expected from the results of the previous
section, there is very little correlation between latent representation of examples encoded by the VAE model (Figure 6 bottom center). Further, the clustering of the example correlations show similar patterns across all models to those described by the preceding analysis.

Figure 6: Comparison of the example activity correlation of each models’ latent representation with hierarchical clustering.

A.5.3 Neuron Activity Correlation

In this final part of the latent representation analysis each row’s values (neurons) of the normalized activity matrices from Figure 5 are correlated with one another for each model. All neurons with no spikes in any example are excluded from this analysis. This allows identification of clusters of active neurons with similar activity patterns across examples.

The neuron activity cluster map of the unregularized SAE model (Figure 7 top left) shows the most prominent clusters of neurons with the strongest correlation coefficients (both positive and negative). The average absolute correlation and clustering is decreasing in the dense SAE (Figure 7 top right), followed by the comparable sparse SAE (Figure 7 top center), AE_{l2} (Figure 7 bottom left), and finally VAE (Figure 7 bottom center). The strong correlation in some models suggests that there is at least some redundant information encoded by different sub-populations of neurons in these models. This redundancy, however, does not seem to improve robustness towards noise in the input images. Finally, the described differences in latent representation qualities need to be considered when using a probabilistic VAE instead of a SAE as a model for unsupervised learning in the brain. [8] states that “a model which does not require neurons to be uncorrelated is desirable because neurons in real cortical networks respond to stimuli in highly correlated ways.”

A.6 Comparison to Earlier Approaches

In this section I aim to relate this work to the two most relevant previous approaches to unsupervised learning in SAEs by [8] and [32].

Network Architecture In her implementation, [8] uses two fully connected layers including feedback connections with LIF neurons with synaptic delay but otherwise unknown temporal dynamics. In contrast to this work, her input layer is also functioning as the output layer of the model. The
hidden representation, however, is not smaller but larger than the input representation as her network has 392 input neurons and 5000 hidden neurons. In contrast to this work and [32], [8] follows Dale’s principle and separates excitatory and inhibitory neurons into distinct populations.

[32] use only convolutional and pooling layers in their SAE implementation. Slightly different networks are used for unsupervised feature learning on MNIST (1 hidden layer) and CIFAR-10 (2 hidden layers). Their kernel size is comparable to the 5x5 filters used in this project, although it is unclear if 5x5 (reported in text) or 7x7 (shown in images) was used. As their MNIST network has 16 and 64 learned filters in their first and second layer respectively, and no fully connected layer in the AE network, the imposed information bottleneck is much weaker compared to this work (16 and 32 filters + fully connected layer with up to 100 neurons in the encoder).

**Image Processing**  
[8] used a specific pre-processing of image data consisting of downscaling and subtraction of the dataset mean of each pixel before presenting images to the network as two separate non-negative “ON” and “OFF” maps. In agreement with this work, both [32] and [8] used a Poisson encoding process to obtain spike sequences from images for network input.

[8] presented the input to her network for a short duration of ca. 10 ms and in a second phase of similar duration without stimulus presentation input layer activity arising from feedback is recorded as network output. This short duration of stimulus presentation is likely a strong limiting factor of model performance. The output activity is substantially lower in spike counts than the input and requires an additional scaling parameter to be fitted before comparing network input to its output. [32] showed each example to their network for 250 ms. Unfortunately, neither [32] nor [8] report how many discrete time steps were simulated in their experiments.

**Learning Rule**  
The learning rule developed by [8] is called mirrored STDP, because the temporal learning dynamics are reversed for excitatory feedforward and feedback connections. How this approach is extended to the inhibitory neurons in her network is not clear. This approach is deemed more biologically plausible than backpropagation, because it requires only local information for weight updates at individual synapses. After learning, network weights are symmetric, which is not necessary in the approach shown in this work.
used a regenerative learning approach based on error backpropagation on the membrane potential similar to this work. In contrast to my approach, however, trained each layer individually and in sequence. For this purpose, an additional pseudo-visible output layer was necessary. The error of this network was calculated based on the precise spike timing, requiring a backpropagation operation after every simulated time step. Why this is relevant for the reconstruction of static images with randomized spike generation process is not clear.

In contrast to the presented approach, both and used no mini-batches for training.

Regularization Methods used an additional synaptic scaling parameter for lifetime sparsity regularization of hidden neurons requiring a target activity hyper-parameter to be defined. The scaling parameter, although functionally identical with a typical bias, is not learned, but adjusted by a predefined rule in dependence on each neuron’s recent average activity. This is a different attempt to a similar goal as the inclusion of the loss terms added in this work, although here target activity is not defined explicitly but implicitly. used no regularization method and no bias term.

Results Both and evaluated their SAEs on MNIST and additionally CIFAR-10. trained the fully connected network for two epochs, but the number of dataset sweeps (per layer) for is unknown.

The lack of convolutional layers in Burbank’s network requires each hidden neuron to learn a receptive field across the entire input space without position or size invariance. While this lead to fixed position Gabor-like filters during CIFAR-10 training, this was not the case for MNIST. The resulting image reconstructions are recognizable but of poor quality for both datasets. evaluated image reconstruction by correlating the spike counts of the input/output layer from the stimulus presentation phase with the feedback phase. calculated their normalized MSE different to the standard deep learning approach used in this work. Unfortunately, numerical comparison between these different approaches is therefore not possible.

While image reconstruction on CIFAR-10 was blurry but promising for , reports runaway network excitation which lead to non-interpretable results. In contrast to the spike regularization applied here, synaptic scaling failed to reliably avoid bursting behavior, although the loss term proposed here has yet to be evaluated on a network with feedback connections.

Analysis of the learned hidden representation by showed a distributed population code that neither requires nor enforces hidden unit decorrelation. This feature has been reproduced in this work, but with a different learning algorithm.

After the layer-wise unsupervised hierarchical feature learning, trained a fully connected layer on the hidden representation of their encoder network for supervised learning, and report a 0.92% classification error which is comparable to other state-of-the-art networks. In this work, hierarchical clustering was used instead of a classification of the latent representations, which again does not allow for a numerical comparison.

Conclusion Both and follow the idea to further integrate computational neuroscience and machine learning. They independently develop different approaches for unsupervised feature learning from MNIST and CIFAR-10 images. Unfortunately, several relevant implementation details are unknown (e.g. discrete time step size, software used for implementation experiments). No code is available for either project, no comparison to non-spiking baseline models is conducted, no noise is added to the input data, and no processing times are available for comparison.

The approach presented in this work is certainly more similar to the work by than , based on the use of hierarchically stacked convolutional layers and error backpropagation. However, here I aimed to address some of the shortcomings of their approach. Notably, I succeeded in training multi-layer SAE networks end-to-end directly without complicated data pre-processing. Further, I compared network dynamics and performance to non-spiking baseline models trained on the same task, and improved transparency by using open-source software and making all code publicly available.