EasyChair Preprint

№ 4310

Biologically Inspired Sleep Algorithm for VariationalAuto-Encoders

Sameerah Talafha, Banafsheh Rekabdar, Christos Mousas and Chinwe Ekenna

October 1, 2020

Biologically Inspired Sleep Algorithm for Variational Auto-Encoders

Sameerah Talafha[1], Banafsheh Rekabdar[1], Christos Mousas[2], and Chinwe Ekenna[3]

[1] Southern Illinois University Carbondale, IL, USA
{sameerah.talafha@,brekabdar@cs}.siu.edu
[2] Purdue University, Indiana, USA
cmousas@purdue.edu
[3] University at Albany, New York, USA
cekenna@albany.edu

Abstract. Variational auto-encoders (VAEs) are a class of likelihood-based generative models that operate by providing an approximation to the problem of inference by introducing a latent variable and encoder/decoder components. However, the latent codes usually have no structure, are not informative, and are not interpretable. This problem is amplified if these models need to be used for auxiliary tasks or when different aspects of the generated samples need to be controlled or interpreted by humans. We address these issues by proposing a biologically realistic sleep algorithm for VAEs (VAE-sleep). The algorithm augments the normal training phase of the VAE with an unsupervised learning phase in the equivalent spiking VAE modeled after how the human brain learns, using the Mirrored Spike Timing Dependent Plasticity learning rule. We hypothesize the proposed unsupervised VAE-sleep phase creates more realistic feature representations, which in turn lead to increase a VAE's robustness to reconstruct the input. We conduct quantitative and qualitative experiments, including comparisons with the state-of-the-art on three datasets: CelebA, MNIST, and Fashion-MNIST. We show that our model performs better than the standard VAE and varitional sparse coding (VSC) on benchmark classification task by demonstrating improved classification accuracy and significantly increased robustness to the number of latent dimensions. As a result of experiments suggest, the proposed method shows improved performance in comparison with other widely used methods and performs favorably under the metrics PSNR, SSIM, LPIPS. The quantitative evaluations also suggest that our model can generate more realistic images compared to the state of arts when tested on disturbed or noisy inputs.

Keywords: Variational Auto Encoder · Spiking Neural Network · Sleep Algorithm.

1 Introduction

Deep neural networks (DNNs) perform well at solving problems that would be prohibitive for human or statistical standards to data classification. Recently, realistic image generation using generative DNNs has been at the forefront of research in machine learning and computer vision [1]. Variational auto-encoder (VAE) [2] is a generative model used to reconstruct training inputs and create random samples from its

learned representations. VAEs offer an efficient way of performing approximate posterior inference with otherwise intractable generative models and yield probabilistic encoding functions that can map complex high-dimensional data to lower-dimensional representations [3]. However, traditional VAEs usually produce latent codes that are not interpretable. This makes the results unsuitable for auxiliary tasks (for example, clustering, segmentation, and classification) and human interpretation. This is contrary to a good generative model as it should not only be able to draw samples from the distribution of data being modeled but also be useful for inference [4]. Moreover, the performance of the VAEs is significantly worsened when the inputs are noisy, rendering them ineffective for the auxiliary tasks [5]. Spiking neural networks (SNNs) are the third generation of neural networks, in which neurons communicate through binary signals known as spikes. SNNs are energy-efficient than DNNs making them suitable for hardware implementation because spikes bring the opportunity of using event-based hardware as well as simple energy-efficient accumulators instead of complex energy-hungry multiply-accumulators that are usually employed in DNNs hardware [6]. SNNs have been proven to be less prone to performance deterioration to noisy inputs than their DNNs counterparts [7]. The spatio-temporal capacity of SNNs makes them potentially stronger than DNNs; however, harnessing their ultimate power is not straightforward [8]. Recently, mimicking biologically inspired learning in VAE has been demonstrated using the Variational Sparse Coding (VSC) model [9], which modeled sparsity in the latent space of VAE with a Spike and Slab prior distribution resulting in latent codes with improved sparsity and interpretability. However, the approach mentioned above falls short in mimicking biologically realistic learning compared to approaches like spike-timing-dependent plasticity (STDP) [10]. Sleep mechanism is essential to several brain functions of humans and animals, including how neurons communicate with each other. During the sleep phase, there is the reactivation of neurons involved in a previously learned activity and this reactivation is likely to invoke the same spatio-temporal pattern as the pattern observed during training in the awake stage [11]. In neuroscience, it is hypothesized that sleep can improve memory, learning, increase attention, and robustness against noise in both humans and animals. In this work, we employ the notion of sleep from biology and apply an off-line unsupervised "sleep" stage to modify the parameters of a fully connected VAE. Our model combines ideas from VAE [2] and the sleep mechanism [12] leveraging the advantages of deep and spiking neural networks (DNN–SNN). During the sleep stage, sleep functions' choice of Mirrored STDP rules [13] increases a subject's ability to form logical connections between memories and to generalize knowledge learned during the awake stage. We hypothesize that sleep could aid in reducing an auto-encoder loss function hence improving VAE output interpretability and producing latent codes that are less dispersed. To the best of our knowledge, this work provides a step towards mimicking the biological sleep stage in the context of learning in generative models like variational auto-encoder. Our contributions are summarized below:

1. We report positive results for image reconstruction datasets (MNIST, CelebA, and Fashion-MNIST) where following sleep the generation produces examples that are more distinct than the original input than before VAE-sleep.

2. We illustrate that our VAE-sleep algorithm creates latent codes which hold a high level of information about our input (image) compared to the standard VAE [2] and VSC [9].
3. We illustrate that our model has a more robust architecture whereby performance on noisy inputs is higher compared to the standard VAE [2] and VSC [9].

The rest of the paper is organized as follows: in section 2 we review related work, section 3 describes our proposed model, 4 details the evaluation, experimental design, and results. Section 5 concludes the paper with remarks on the obtained results.

2 BACKGROUND AND RELATED WORK

2.1 Varitional Auto-Encoder (VAE)

VAE is an unsupervised model with efficient coding with goals to maximize the marginal likelihood $\prod p(x_i)$ with respect to the decoding parameter θ of the likelihood function $p_\theta(z|x)$ and the encoding parameter ϕ of the recognition model $q_\phi(z|x)$. The input $x_i \in \mathcal{R}^{M\times 1}$ is passed to the encoder network, producing an approximate posterior $q_\phi(z|x)$ over latent variables z. The sample $z_i \in \mathcal{R}^{J\times 1}$ is drawn from a prior $p(z)$ which can be chosen to take different parametric forms. In most types of VAEs, the prior takes the form of a multivariate Gaussian with identity covariance $\mathcal{N}(z; 0, I)$. For example, if q were Gaussian, it would be the mean and variance of z for each data point $\phi_{x_i} = (\mu_{x_i}, \sigma^2_{x_i})$. Then, z_i is passed through the feedforward decoder network to compute the probability of the input $p_\theta(x|z)$ given the sample. The $p_\theta(x|z)$ is chosen to fit the expected nature of variation in the observations. The key step in the derivation of VAE's loss function is the definition of a lower bound on the log-likelihood $\log p_\theta(x)$, referred as the **E**vidence **L**ower **BO**und (ELBO) that depends on $q_\phi(z|x)$ [9]. Based on Jensen's inequality, the ELBO can be formulated as

$$\log p_\theta(x_i) = \int p_\theta(x_i|z)p(z)\frac{q_\phi(z|x)}{q_\phi(z|x)}dz \geqslant \mathcal{L}_{\theta,\phi;x_i}, \tag{1}$$

$$\mathcal{L}_{\theta,\phi;x_i} = \mathbb{E}_{q_\phi(z|x_i)}\left[\log p_\theta(x_i|z)\right] - \mathcal{D}_{KL}(q_\phi(z|x_i)\|p_\theta(z)). \tag{2}$$

ELBO is divided into two terms; the first term is the reconstruction likelihood that maximizes the expectation of the data likelihood under the recognition function. The second term is the Kullback–Leibler (KL) that ensures the learned distribution q is similar to the true prior distribution p, as shown in the following.The value of ELBO is tight when the difference between approximate and true posterior is zero, $q_\phi(z|x) = p_\theta(z|x)$, and the tightness of the bound depends on the KL divergence of $q_\phi(z|x)$ and $p_\theta(z|x)$. The images generated by VAEs are usually not realistic and blurry [4]. Using Convolutional Neural Networks (CNN) improves the performance of VAE by capturing important perceptual features such as spatial correlation [14], but the fidelity and naturalness of reconstruction are still unsatisfactory [9].

2.2 Varitional Sparse Coding (VSC)

VSC approach comprises of increasing sparsity in the latent space of VAE, representing it as a binary spike and Slab probability density function (PDF) [9]. It employs the non-linear features that constitute variability in data and exemplifies them as few non-zero elements in sparse vectors. Let s_j denote a binary spike and z_j a continuous slab variable. s_j value is either one or zero with defined probabilities α and $(1-\alpha)$ respectively and z_j distribution is either a Gaussian or a Delta function centered at zero, conditioned on whether s_j is one or zero respectively. As VAE, $p_\theta(x|z)$ denote a probabilistic decoder with a neural network to generate observed data x from sparse vectors in the latent space. The prior probability density $p(z_s)$ over the latent variable z_i is defined below:

$$p_s(z) = \prod_{j=1}^{J} \left(\alpha \mathcal{N}(z_j; 0, 1) + (1-\alpha)\delta(z_j)\right), \tag{3}$$

Where $\delta(\cdot)$ indicates the Dirac delta function centered at zero. The distribution of representation corresponding to the x approximated by the variational posterior $q_\phi(z|x)$, which is produced by an encoder with a neural network of the form

$$q_\phi(z|x_i) = \prod_{j=1}^{J} \left(\gamma_{i,j}\mathcal{N}(z_{i,j}\mu_{z,i,j}\sigma^2_{z,i,j}) + (1-\gamma_{i,j})\delta(z_{j,j})\right), \tag{4}$$

Where $\mu_{z,i,j}$ is the mean, $\sigma^2_{z,i,j}$ is the variance, and $\gamma_{i,j}$ is a Spike probabilities vector constrained between 0 and 1 . They are the outputs of the recognition function $p(z|x_i)$ composed of a neural network which takes as input an observation x_i. The $p(z|x_i)$ allows for the posterior to match the prior and allows the freedom to control the Gaussian moments and the Spike probabilities independently enable the model to encode information in the latent space. In contrast to equation 3, which represents standard Slab and Spike distribution, equation 4 describes distribution of Slab variables having Gaussian distributions $\mathcal{N}(z_{i,j}; \mu_{z,i,j}, \sigma^2_{z,i,j})$ and Spike variables having probabilities of being one $\gamma_{i,j}$. VSC's loss function introduces a sparsity KL divergence penalty term with the two terms of ELBO, hence modifying equation 2 as:

$$\begin{aligned}\mathcal{L}_{\theta,\phi;x_i} = \mathbb{E}_{q_\phi(z|x_i)}\left[\log p_\theta(x_i|z)\right] + \underset{\theta,\phi,w,x_u}{\arg\max} \sum -\mathcal{D}_{KL}(q_\phi(z|x_i)\|p(z)) \\ -J \cdot \mathcal{D}_{KL}(\bar{\gamma}_{u*}\|\alpha),\end{aligned} \tag{5}$$

where $\bar{\gamma}_{u*}$ is the average Spike probability of each pseudo input recognition model that matches with the prior sparsity α in the latent space. Despite its utility to improve the performance of VAE model, the implementation is not biologically realistic [9].

3 Proposed Model

3.1 Spiking Varitional Auto-Encoder (SVAE)

While DNNs' cell is designed to simulate highly simplified brain dynamics, SNNs' cell aims to closely model temporal brain dynamics. SNNs are combining digital-analog

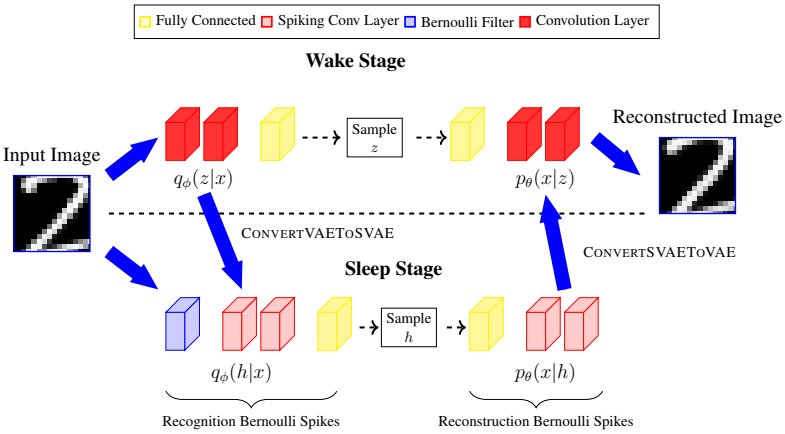

Fig. 1: The block diagram of our proposed model.

machines that use the temporal dimension, not just as a neutral substrate for computing, but as a means to encode and process information. Training for probabilistic models of SNNs has recently been investigated in variational inference principles based on the unsupervised mechanism of STDP which updates synaptic weights based on local input and output spikes [12]. In this section, we present the details of our spiking varitional auto-encoder (SVAE). We propose to use mirrored spike-timing-dependent plasticity (Mirrored STDP) [13] instead of STDP [10], since Mirrored STDP follows a learning rule that approximately minimizes auto-encoder loss function compared to STDP. STDP changes in feedforward synaptic strength, but it does not cause the correct changes for the feedback synapses. On the other hand, under Mirrored STDP the plasticity due to any pair of visible and hidden spikes will be the same for the feedforward connections as for the feedback connections, up to a scaling factor. The Mirrored STDP learning rule is given by:

$$\Delta w_{ji} = \begin{cases} a_{LTP} \times (w_{ji} - w_{LB}) \times (w_{UP} - w_{ji}) & t_i - t_j \leq 0, \\ a_{LTD} \times (w_{ji} - w_{LB}) \times (w_{UP} - w_{ji}) & t_i - t_j > 0, \end{cases} \tag{6}$$

where j and i refer to the hidden- and visible-synaptic neurons, respectively, Δw_{ji} is the amount of weight change for the synapse connecting the two neurons, and a_{LTP}, and a_{LTD} scale the magnitude of weight change. Besides, $(w_{ji} - w_{LB}) \times (w_{UP} - w_{ji})$ is a stabilizer term which slows down the weight change when the synaptic weight is close to the weight's lower w_{LB} and upper w_{UB} bounds. We use simulated networks of leaky integrate-and-fire (LIF) neurons in the experiments, which is the most popular one for building SNNs. LIF neurons are characterized by the internal state called the membrane potential. The membrane potential integrates the inputs over time and generates an output spike when the neuronal firing threshold. The objective of SVAE learning is to find weights such that the reconstruction closely matches the original stim-

ulus input, thus ensuring that the hidden unit representation is a good one; intuitively, reconstructions can only be accurate when the hidden layer retains sufficient information about the visible layer. Each observation (in our case, an image), $x^m \in \{n_v \times T\}$ with $m = [1, ..., M]$ in the training set is a collection of n_v pixels of T binary samples with value "1" representing a spike. Parameter n_v is hence the number of observed, or visible spike trains. All M examples in the training set $\mathcal{X} = \{x^m\}_{m=1}^{M}$ are conventionally assumed to be independent and identically distributed (i.i.d.) according to the given true data distribution. SVAE's decoder is a generative probabilistic SNN whose behavior is defined by a parameterized joint distribution $p_\theta(x, h)$ over visible spiking signals $x = [x_1, ..., x_{n_v}]$ and hidden spiking signals $h = [h_1, ..., h_{n_h}]$. The joint distribution of x and h is modeled as:

$$p_\theta(x, h) = p_\theta(h)p_\theta(x|h) = \prod_{j=1}^{n_h} p_{\theta_j}(h_j) \prod_{i=1}^{n_v} p_{\theta_i}(x_i|h), \tag{7}$$

where parameter n_h is hence the number of hidden spike trains, and θ is the vector of parameters that define the prior distribution $p_\theta(h)$ of the latent spikes and the conditional distribution $p_\theta(x|h)$. Each the hidden spiking signal (latent spike) has Bernoulli samples distributed as:

$$p_{\theta_j}(h_j) = \prod_{t=1}^{T} F\left(\left(2h_{j,t} - 1\right)\theta_j\right), \tag{8}$$

where θ_j is the prior log-likelihood ratio for every sample $h_{j,t} \in 0, 1$, and $F(x)$ is the ReLU function. Since x is conditionally independent to h, the conditional distribution is as shown in equation 9:

$$p_{\theta_i}(x_i|h) = \prod_{t=1}^{T} F\left(\left(2x_{i,t} - 1\right)u_{i,t}\right), \tag{9}$$

where $u_{i,t}$ is the membrane potential of the i-th visible neuron at time t. The $u_{i,t}$ evolves over time as a dynamic system that depends on the past spiking behaviour of the hidden and visible neuron i, as explained next. Assuming a feedforward synaptic memory of τ_α samples and a feedback memory of τ_β samples, the membrane potential is as shown in equation 10:

$$u_{i,t} = \sum_{j=1}^{n_h} \alpha_{j,i}^T h_{j,t-\tau_\alpha}^{-1} + \beta_i^T x_{i,t-\tau_\beta}^{t-1} + \gamma_i, \tag{10}$$

where $\alpha_{j,i} \in \mathcal{R}^{\tau_\alpha \times 1}$ is the kernel for the synapse between hidden neuron j and visible neuron i, and $\beta_{j,i} \in \mathcal{R}^{\tau_\beta \times 1}$ is the feedback filter for dynamic spikes of neuron i. We model the feedforward and feedback filters as the linear combination of $K_\alpha \leqslant \tau_\alpha$ and $K_\beta \leqslant \tau_\beta$ basis functions, respectively:

$$\alpha_{j,i} = Aw_{j,i} = \sum_{k=1}^{K_\alpha} w_{j,i,k} a_k, \tag{11}$$

$$\beta_{j,i} = Bv_{j,i} = \sum_{k=1}^{K_\beta} v_{j,i,k} b_k, \tag{12}$$

where $A = [a_1, ..., a_{K_\alpha}]$ and $B = [b_1, ..., b_{K_\beta}]$ are the basis vectors, $w_{j,i} = [w_{j,i,1}, ..., w_{j,i,K_\alpha}]^T$ and $v_{j,i} = [v_{j,i,1}, ..., v_{j,i,K_\beta}]^T$ are learnable weights. SVAE's encoder creates output spiking signals h according to the variational distribution $q_\phi(h|x)$ given the visible spiking signals x as.

$$q_\phi(h|x) = \prod_{j=1}^{n_h} \prod_{t=1}^{T} F\left((2h_{j,t} - 1)\,\tilde{u}_{j,t}\right), \tag{13}$$

where $\tilde{u}_{j,t}$ is the varitional membrane potential given as:

$$\tilde{u}_{i,t} = \sum_{i=1}^{n_v} \tilde{w}_{j,i}^T A^T x_{i,t+1}^{t+\tau_\alpha} + \tilde{v}_{j,i}^T B^T h_{j,t+1}^{t+\tau_\beta} + \tilde{\gamma}_i. \tag{14}$$

The gradient of the ELBO for x with respect to the model parameters θ and ϕ can be calculated as:

$$\nabla_\theta \mathcal{L}_{\theta,\phi}(x) = \mathbb{E}_{q_\phi(h|x)}\left[\nabla_\theta \log p_\theta(x,h)\right], \tag{15}$$

$$\nabla_\phi \mathcal{L}_{\theta,\phi}(x) = \mathbb{E}_{q_\phi(h|x)}\left[l_\phi(x,h)\nabla_\phi, \log q_\phi(h|x)\right], \tag{16}$$

where,

$$l_\phi(x,h) = \log p_\theta(x,h) - \log q_\phi(x|h). \tag{17}$$

3.2 VAE-Sleep

Algorithm 1 shows the pseudo-code for our VAE-sleep algorithm. In the wake stage, We train the VAE using stochastic gradient descent to optimize the loss with respect to the encoder's parameters and decoder θ and ϕ with backpropagation by applying re-parameterization trick [2]. In CONVERTVAETOSVAE function, we assume the VAE utilizes ReLU neurons with no bias in each layer. This assumption is made so that the output neuron's activation can be treated as a firing rate, either zero or positive, and the thresholds of all LIF neurons in SVAE in a given layer are of the same scale. Therefore, the weights from the VAE are directly mapped to the SVAE as in [16]. After training, the network structure is converted into SVAE. Next, we run a sleep stage in which inputs to each neuron of the visible layer must be presented as Bernoulli spike-trains to propagate activity from the visible layer to the hidden layers of the network. For that, we convert inputs (real-valued pixel intensities or features) to Bernoulli spike-trains using the Berniolli Filter [17]. At each iteration, we feed the SVAE architecture (the encoder followed by the decoder) with input data. The neurons in SVAE only fire termed as 'spikes', when the neurons reach a certain $threshold$ membrane potential.

Algorithm 1 VAE-Sleep

procedure CONVERTVAETOSVAE(vae)
Map the weights from (vae) with ReLU units to network integrate-fire units
Apply weight normalization [15] and return scales for each layer in encoder and decoder
return $svae, scales$
end procedure
procedure CONVERTSVAETOVAE($svae$)
Directly map the weights from LIF neurons in SNN layer ($svae$) to ReLU neuron (vae) in DNN layer **return** (vae)
end procedure
procedure SLEEP($save$, I, $scales$) ▷ I is input
Initialize u (voltage) $= 0$ vectors for all neurons
for $t \leftarrow 1, Ts$ **do** ▷ Ts – sleep duration
$S(1) \leftarrow$ Convert input I to Bernoulli-distributed spike train
for $l \leftarrow 2, n$ **do** ▷ n – number of layers
$u(l,t) \leftarrow u(l,t-1) + (scales(l-1)\mathbf{W}(\mathbf{l},\mathbf{l-1})S(l-1))$ ▷ $W_{l,l-1}$ – weights
$S(l,t) \leftarrow u(l,t) > threshold(l)$
$$\mathbf{W}(\mathbf{l},\mathbf{l-1}) \leftarrow \begin{cases} \mathbf{W}(\mathbf{l},\mathbf{l-1}) + \Delta\mathbf{W} \text{ if } S(l) = 1 \ \& \ S(l-1) = 1 \\ \mathbf{W}(\mathbf{l},\mathbf{l-1}) - \Delta\mathbf{W} \text{ if } S(l) = 1 \ \& \ S(l-1) = 0 \end{cases}$$ ▷ *mSTDP
end for
end for
end procedure
procedure MAIN
Initialize neural netowrk (vae) with ReLU neurons and bias $= 0$.
Train vae using backpropgation with respect θ and ϕ by applying re-parameterization trick
$svae, scales =$ CONVERTVAETOSVAE(vae)
$svae =$ SLEEP($svae$, Training data X, $scales$)
$vae =$ CONVERTSVAETOVAE($svae$)
end procedure
*mSTDP: Mirrored Spike Timing Dependent Plasticity

We compare the encoded-decoded output with the input data and gradient of the ELBO with respect θ and ϕ, the error through the architecture and the Mirrored STDP rule is applied to update weights. After the sleep stage, the SVAE network is converted back into the VAE (CONVERTSVAETOVAE), and testing is performed. The block diagram of our proposed model is shown in Fig. 1.

4 Experiment Results and Dataset

We benchmark our model using the following datasets commonly used for evaluating VAEs' performance: (a.) MNIST [18], (b.) Fashion-MNIST [19], and (c.) CelebA [20].We compared the performance of our SVAE model with the performance of VAE and VSC implemented in [9]. The VAE model is a Convolutional Variational auto-encoder (CVAE). The detailed architecture of CVAE is as described in Table 1. To establish a valid benchmark, we implemented the VSC architecture in the same way as the

VAE model. Table 2 summarizes the parameters used for training the VAE and the VSC models. Our proposed model (Fig. 1) consists of two stages: wake and sleep. For the wake stage, a CVAE model is used. The architecture of the CVAEs' encoder consisted of 2 convolutional layers for MNIST and Fashion-MNIST datasets and 4 layers for the CelebA, each with 32 channels, 4×4 kernels, and a stride of 2. This was followed by 2 fully connected layers, each of 256 units. The latent distribution consisted of one fully connected layer of the mean and log standard deviation of Gaussian random variables. The decoder architecture is simply the transpose of the encoder, but with the output parametrizing the Bernoulli distributions over the pixels. The sleep stage (section 3.2), is implemented as the SVAE model explained in section (3.1). SVAE constitutes spiking convolution layers with LIF neurons to replicate convolution layers in the VAE model used during the wake stage. The parameters used for training the SVAE are shown in Table 4. To evaluate the robustness of our method on noisy inputs, we create a noisy test set. In this test set, we distort the inputs by adding noise vectors sampled from a standard normal distribution (Gaussian noise) to the test image encodings.

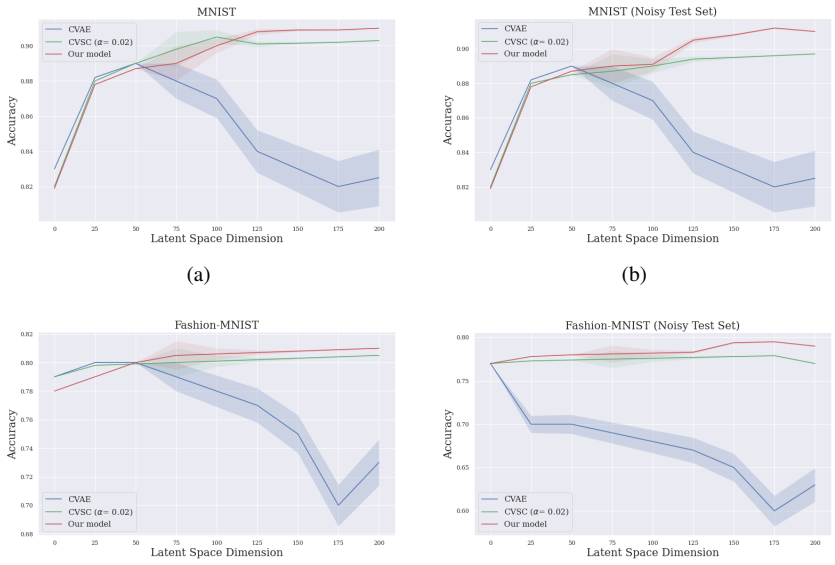

Fig. 2: Classification performance of our proposed model, VSC, and VAE at varying number of latent space dimensions for MNIST and Fashion-MNIST datasets vs noisy MNIST and noisy Fashion-MNIST datasets (distorted with gaussian noise).

4.1 Quantitative Evaluation

We used the following metrics – Peak Signal to Noise Ratio (PSNR) [21], Structural Similarity Index (SSIM) [22], and Learned Perceptual Image Patch Similarity (LPIPS)

Table 1: CVAEs' Encoder and Decoder architecture details.

Dataset	Encoder	Decoder
CelebA	Input $64 \times 64 \times 3$ RGB image	Input: the latent variable z
	4×4 conv. 32 ReLU. stride 2	FC. 256 ReLU
	4×4 conv. 32 ReLU. stride 2	FC.$4 \times 4 \times 64$ ReLU
	4×4 conv. 64 ReLU. stride 2	4×4 deconv. 64 ReLU. stride 2
	4×4 conv. 64 ReLU. stride 2	4×4 deconv. 32 ReLU. stride 2
	FC. 256.	4×4 deconv. 32 ReLU. stride 2
	FC latent-sz (Mean)‖FC latent-sz, Sigmoid (Std. dev.)	4×4 deconv. 3. stride 2
MNIST/Fashion-MNIST	Input $28 \times 28 \times 1$ grey-scale image	Input : the latent variable z
	4×4 conv. 32 ReLU. stride 2	FC. 256 ReLU
	4×4 conv. 32 ReLU. stride 2	FC.$4 \times 4 \times 32$ ReLU
	FC. 256.	deconv. 32 ReLU. stride 2
	FC latent-sz (Mean) ‖FC latent-sz, Sigmoid (Std. dev.)	4×4 deconv. 1. stride 2

Table 2: Parameters used for training VAE and VSC; hidden-sz: number of hidden neurons, latent-sz: number of latent dimensions, Epochs: number of training epochs, lr: learning rate, log-interval: number of batches before logging training status, β: adjustable hyperparameter to balance latent channel capacity and independence constraints with reconstruction accuracy, c: stability of the gradient ascent.

Dataset	hidden-sz	latent-sz	Epochs	lr	log-interval	normalize	β	$\Delta\beta$	α	c	Δ c	iterations
CelebA	256	400	500	$3e-4$	500	False	4	0	$1e-2$	50	$1e-3$	$20,000$
MNIST/Fashion-MNIST	256	200	100	$1e-4$	500	False	2	0	$1e-2$	50	$1e-3$	$20,000$

[23] to evaluate the quality of the generated images and learning in the latent space [9] as well as to evaluate the quality of the latent codes which affect the performance of auxiliary tasks like classification. PSNR is a standard error metric used to compare image compression quality defined as in equation 18:

$$\text{PSNR} = -10 \log \left(\frac{(2^n - 1)^2}{\text{MSE}} \right), \tag{18}$$

where MSE is the Mean Square Error between the ground truth image X and the generated image Y as in equation 19:

$$\text{MSE} = \frac{1}{\text{HW}} \sum_{i=1}^{\text{H}} \sum_{j=1}^{\text{W}} |X_{i,j} - Y_{i,j}|, \tag{19}$$

Table 3: Parameters used for training SVAE. Input rate: maximum firing rate of the input neurons, Sleep duration: length of sleep (number of images presented during sleep stage), Thresholds: neuronal firing thresholds for each layer of neurons, Synaptic AMPA current: scale factor of weights during sleep stage, upper-bound: weights' upper bound range, lower-bound: weights' lower bound range, ρ: target activation rate.

Dataset	Input Rate	Sleep Duration	Thresholds	Synaptic AMPA current	upper-bound	lower-bound	ρ
CelebA	500Hz	200600	15,40,23,0.9	2.19	0.013	0.019	0.02
MNIST/Fashion-MNIST	40 Hz	48000	36.18, 23.36	2.19	0.0063	0.0069	0.02

SSIM is used to evaluate differences of brightness, contrast, and image structure (it's range: $[0, 1]$) defined as equation 20:

$$SSIM(X, Y) = \frac{(2\mu_{X,Y}\mu_Y + C_1)(2\sigma_{XY} + C_2)}{(\mu_X^2 + \mu_Y^2 + C_1)(\sigma_X^2 + \sigma_Y^2 + C_2)}, \tag{20}$$

where μ_X and μ_Y denote the mean values of images X and Y, σ_X and σ_Y denote the variances of X and Y, σ_{XY} is the covariance between X and Y, and C is a constant.

In contrast to SSIM and PSNR which lean towards preference of blurry predictions over sharper but imperfect generations, LPIPS has a better correlation to human judgment. Given two images [24] LPIPS returns a weighted L_2 distance (image differences) in the space of hidden unit when run through the convolutional part of an image classification network such as Visual Geometry Group (VGG). Table 4 (left hand side) shows the quantitative evaluation of the generation results for the metrics mentioned above. Generally, VAE generates distorted images when tested with noisy test sets [5], so, in order to evaluate the effect of the noise, we test VAE, VSC, and our model with noisy test sets. Table 4 (right hand side) presents the quantitative evaluation of the generation results for the noisy test sets. We report the average PSNR, SSIM, and LPIPS of the closet samples to the ground truth for each test image. Our model outperforms the VAE and VSC on all metrics by a significant margin. Learning in the latent space measures the ability of a model to recover latent codes that hold a high level of information about the input by performing a standard classification experiment using the latent variables as input features. Fig. 2 shows the classification performance obtained on the test sets MNIST and Fashion-MNIST, noisy MNIST, and noisy Fashion-MNIST. As Fig. 2 presents, VAE reaches its peak performance for the optimal choice of latent space dimensions, but yields inefficient codes if the latent space is too large. However, the performance of our model and VSC is independent of latent space dimensions, making them able to reliably recover efficient codes without the need to specify an optimal latent space size. Additionally, our model outperforms VSC for noisy test images.

Table 4: Quantitative evaluation for test set with and without noise.

		test set without noise			**noisy test set**		
Dataset	Model	PSNR (↑)	SSIM (↑)	LPIPS -VGG (↓)	PSNR (↑)	SSIM (↑)	LPIPS-VGG (↓)
CelebA	CVAE	20.9 dB	0.620	0.021	16.70 dB	0.430	0.034
	CVSC	28.9 dB	0.789	0.014	24.10 dB	0.700	0.023
	Our Model	**29.6 dB**	**0.800**	**0.013**	**27.20 dB**	**0.750**	**0.019**
MNIST	CVAE	33.00dB	0.833	0.017	29.00dB	0.659	0.025
	CVSC	34.40dB	0.960	**0.012**	30.40dB	0.870	0.021
	Our Model	**34.57dB**	**0.965**	**0.012**	**32.57dB**	**0.910**	**0.014**
Fashion -MNIST	CVAE	30.30dB	0.590	0.019	21.65dB	0.430	0.027
	CVSC	30.55dB	0.760	0.014	26.97dB	0.680	0.022
	Our Model	**31.07dB**	**0.800**	**0.015**	**28.97dB**	**0.710**	**0.018**

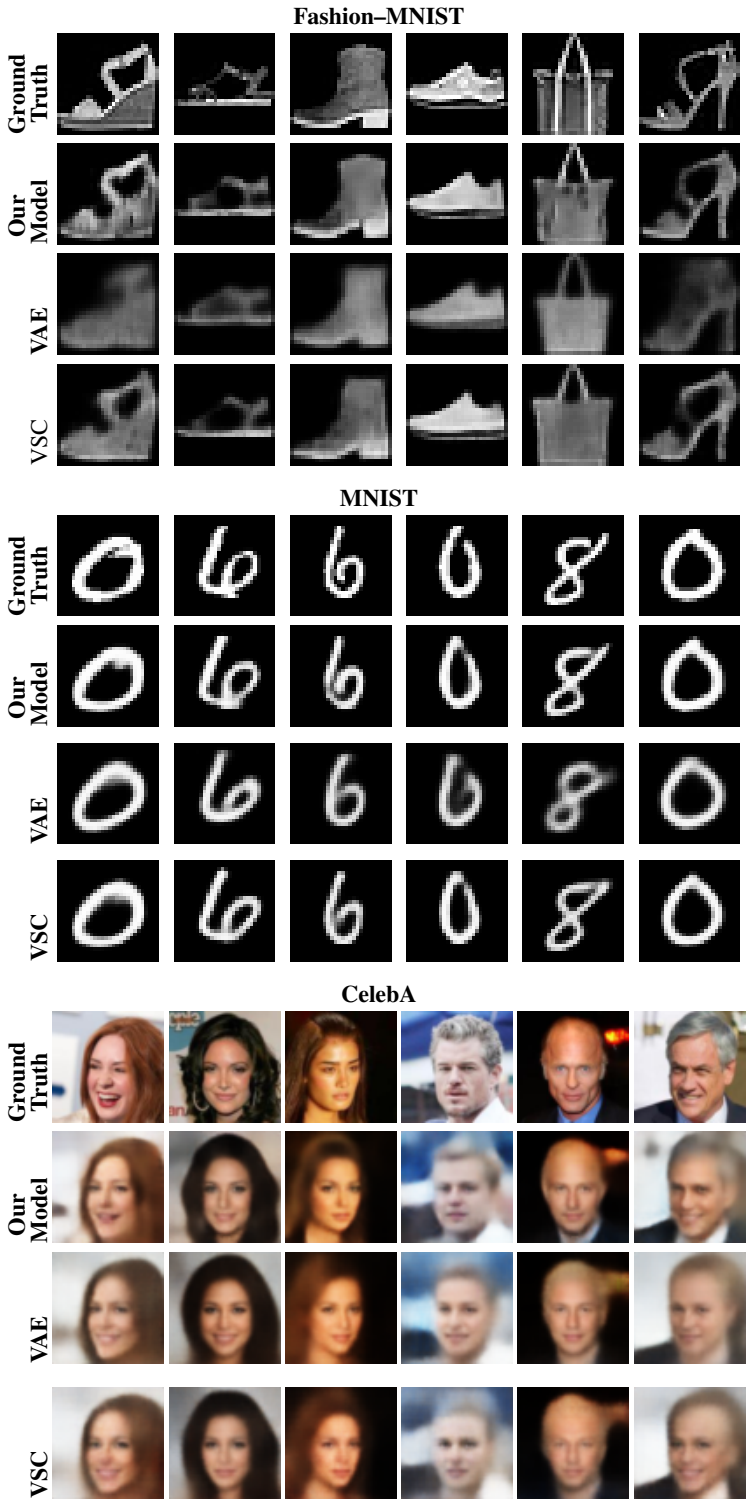

Fig. 3: Examples of image reconstruction for the CelebA, MNIST, and Fashion-MNIST datasets.

4.2 Qualitative Evaluation

Fig. 3 shows the qualitative comparisons between our proposed model, VAE and VSC for three different datasets. As Fig. 3 presents the samples generated by our model more closely resemble the ground truth compared with the other models, hence proving our hypothesis that using sleep algorithm on VAE (VAE–sleep algorithm) improves output interpretability and producing latent code that is less dispersed. The most noticeable distinction in the quality of the generated images can be seen in the output of our model vs VSC and VAE for the CelebA dataset (Fig. 3), not only the images are more clear, but also, our model is able to retain and replicate facial features in more details.

5 Conclusion and Future work

In this work, we propose a biologically realistic sleep algorithm for VAEs. The algorithm augments the normal training phase of the VAE with an unsupervised learning phase in the equivalent SVAE modeled after how the human brain learns, using the Mirrored STDP rules. We hypothesize that the unsupervised VAE-sleep phase creates more natural feature representations, which leads to increase a VAE's robustness to reconstruct the input. We show that our model performs better than the standard VAE and VSC on benchmark classification task (MNIST and Fashion-MNIST) by demonstrating improved classification accuracy and significantly increased robustness to the number of latent dimensions. The quantitative evaluations show that our model can generate more realistic images compared to the state of arts when tested on disturbed or noisy inputs. To the best of our knowledge, this work provides a step towards mimicking biological sleep stage in the context of learning in generative models like VAE. Future work includes: improving spike encoding [17] used in the spiking convolution layer, scaling the network to perform experiments on images with a larger size than we currently experiment with, and applying the sleep stage to a generative model for a spatio-temproal dataset like video.

References

1. S. Dreiseitl and L. Ohno-Machado, "Logistic regression and artificial neural network classification models: a methodology review," *Journal of biomedical informatics*, vol. 35, no. 5-6, pp. 352–359, 2002.
2. H. Kim and A. Mnih, "Disentangling by factorising," *arXiv preprint arXiv:1802.05983*, 2018.
3. A. Mishra, S. Krishna Reddy, A. Mittal, and H. A. Murthy, "A generative model for zero shot learning using conditional variational autoencoders," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 2188–2196.
4. P. E. Brown, G. O. Roberts, K. F. Kåresen, and S. Tonellato, "Blur-generated non-separable space–time models," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 62, no. 4, pp. 847–860, 2000.
5. S. S. Roy, M. Ahmed, and M. A. H. Akhand, "Noisy image classification using hybrid deep learning methods," *Journal of Information and Communication Technology*, vol. 17, no. 2, pp. 233–269, 2018.

6. A. Ankit, A. Sengupta, P. Panda, and K. Roy, "Resparc: A reconfigurable and energy-efficient architecture with memristive crossbars for deep spiking neural networks," in *Proceedings of the 54th Annual Design Automation Conference 2017*, 2017, pp. 1–6.
7. T. Iakymchuk, A. Rosado-Muñoz, J. F. Guerrero-Martínez, M. Bataller-Mompeán, and J. V. Francés-Víllora, "Simplified spiking neural network architecture and stdp learning algorithm applied to image classification," *EURASIP Journal on Image and Video Processing*, vol. 2015, no. 1, p. 4, 2015.
8. M. Mozafari, M. Ganjtabesh, A. Nowzari-Dalini, and T. Masquelier, "Spyketorch: Efficient simulation of convolutional spiking neural networks with at most one spike per neuron," *Frontiers in neuroscience*, vol. 13, 2019.
9. F. Tonolini, B. S. Jensen, and R. Murray-Smith, "Variational sparse coding," in *Uncertainty in Artificial Intelligence*. PMLR, 2020, pp. 690–700.
10. M. Mozafari, M. Ganjtabesh, A. Nowzari-Dalini, S. J. Thorpe, and T. Masquelier, "Bio-inspired digit recognition using reward-modulated spike-timing-dependent plasticity in deep convolutional networks," *Pattern Recognition*, vol. 94, pp. 87–95, 2019.
11. M. A. Wilson and B. L. McNaughton, "Reactivation of hippocampal ensemble memories during sleep," *Science*, vol. 265, no. 5172, pp. 676–679, 1994.
12. T. Tadros, G. Krishnan, R. Ramyaa, and M. Bazhenov, "Biologically inspired sleep algorithm for increased generalization and adversarial robustness in deep neural networks," in *International Conference on Learning Representations*, 2019.
13. K. S. Burbank, "Mirrored stdp implements autoencoder learning in a network of spiking neurons," *PLoS computational biology*, vol. 11, no. 12, p. e1004566, 2015.
14. Y. Pu, Z. Gan, R. Henao, X. Yuan, C. Li, A. Stevens, and L. Carin, "Variational autoencoder for deep learning of images, labels and captions," in *Advances in neural information processing systems*, 2016, pp. 2352–2360.
15. T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," in *Advances in neural information processing systems*, 2016, pp. 901–909.
16. B. Rueckauer and S.-C. Liu, "Conversion of analog to spiking neural networks using sparse temporal coding," in *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2018, pp. 1–5.
17. H. Hazan, D. J. Saunders, H. Khan, D. Patel, D. T. Sanghavi, H. T. Siegelmann, and R. Kozma, "Bindsnet: A machine learning-oriented spiking neural networks library in python," *Frontiers in neuroinformatics*, vol. 12, p. 89, 2018.
18. Y. LeCun, "The mnist database of handwritten digits," *http://yann. lecun. com/exdb/mnist/*, 1998.
19. H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.
20. Z. Liu, P. Luo, X. Wang, and X. Tang, "Large-scale celebfaces attributes (celeba) dataset," *Retrieved August*, vol. 15, p. 2018, 2018.
21. Q. Huynh-Thu and M. Ghanbari, "Scope of validity of psnr in image/video quality assessment," *Electronics letters*, vol. 44, no. 13, pp. 800–801, 2008.
22. Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli *et al.*, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.
23. R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 586–595.
24. X. Zhang, J. Zou, K. He, and J. Sun, "Accelerating very deep convolutional networks for classification and detection," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 10, pp. 1943–1955, 2015.