Introduction :

Spiking neural networks (SNNs) can potentially offer an efficient way of doing inference because the neurons in the networks are sparsely activated and computations are event-driven. Conversion of popular CNN architectures, including VGG-16 and Inception-v3, into SNNs that produce the best results reported to date on MNIST, CIFAR-10, and the challenging ImageNet dataset. SNNs can trade-off classification error rate against the number of available operations whereas deep continuous-valued neural networks require a fixed number of operations to achieve their classification error rate. From the examples of LeNet for MNIST and BinaryNet for CIFAR-10, we show that with an increase in error rate of a few percentage points, the SNNs can achieve more than 2x reductions in operations compared to the original CNNs. This highlights the potential of SNNs in particular when deployed on power-efficient neuromorphic spiking neuron chips, for use in embedded applications.

Training a Convolutional SVAE deep spiking network (i.e. learning the synaptic weights) is difficult. An alternative approach is to take a pre-trained neural network and convert it into a spiking neural network. In this CVAE-to-CSVAE conversion, we use the weights of the CVAE and replace the analog (rate) neurons of the ANN by simple Integrate-and-Fire spiking neurons. We convert a trained Analog Neural Network (convolutional layer), consisting of Rectified LinearUnits (ReLU), to convolutional SNN composed of integrate-and-fire neurons with “proper” firing thresholds.

The steps (see the digram in the paper):

1) Train the convolution CVAE . (see vae_models.py )
2) We used the CVAE weight as the initial values of CSVAE weights, by mapping the weight in each convolution layer in CVAE with the similar convolution layer in CSVAE. (see svae_models.py)
3) We convert the input images to Bernoulli distribution. (see transforms.py )
4) We run CSVAE.
5) Then we transform the weights from each convolution layer in CSVAE to CVAE to generate the image. (see w_save_to_vae.py )