RV-VAE: Integrating Random Variable Algebra into Variational Autoencoders (Supplementary Material)

Vassilis C. Nicodemou[1,2] Iason Oikonomidis[2] Antonis Argyros[1,2]

[1]Computer Science Department, University of Crete, Heraklion, Greece
[2]Institute of Computer Science, FORTH, Heraklion, Greece
{nikodim, oikonom, argyros}@ics.forth.gr

1. Introduction

This text supplements the main manuscript of the work titled "RV-VAE: Integrating Random Variable Algebra into Variational Autoencoders". The main aim of this material is:

- to clarify the usage of algebra for Random Variables (RVs) on ANN operations,
- to present experiments and empirical evidence that support the assumption according to which in commonly used scenarios, non-linear activation units typically process normally distributed data, and,
- to present more qualitative results obtained from the proposed approach.

2. Random Variable Algebra on ANN Operations

As stated in the main manuscript, all ANN operations can be adapted for the case of RV input using the rules of random variable algebra. In order to comprehend how the rules apply to our case, we must clarify that for every ANN operation, the input to the operation is a RV, while the parameters (weights and biases) are scalars. That is the reason why in Section 3 of the main manuscript, all modifications consider the case of a RV operand and a scalar. Moreover, due to the way these operations are defined, each RV is only summed with other RVs (never multiplied). This observation lays a solid basis for proving the assumption we make regarding normal data distribution.

2.1. Convolutional/transposed convolutional operations

Convolutional operations can be viewed as a special case of general linear operations, where some of the elements of $\boldsymbol{A}$ are forced to be zero. Therefore, the derivation is very similar to the case presented in Section 3.1 of the main manuscript. The convolutional operation is defined as:

$$\boldsymbol{y} = \boldsymbol{x} * \boldsymbol{A} + \boldsymbol{b}, \tag{1}$$

where again $\boldsymbol{x}$ is the input vector of RVs, $\boldsymbol{A}$ the matrix (kernel) of learnable weights, and $\boldsymbol{b}$ the learnable bias vector. The expected value of output $\boldsymbol{y}$ is:

$$\mathbb{E}[\boldsymbol{y}] = \mathbb{E}[\boldsymbol{x} * \boldsymbol{A} + \boldsymbol{b}] = \mathbb{E}[\boldsymbol{x}] * \boldsymbol{A} + \boldsymbol{b} \tag{2}$$

and the variance is:

$$\text{var}[\boldsymbol{y}] = \text{var}[\boldsymbol{x} * \boldsymbol{A} + \boldsymbol{b}] = \text{var}[\boldsymbol{x}] * (\boldsymbol{A} \odot \boldsymbol{A}). \tag{3}$$

A similar procedure is followed to obtain the expected value and variance of the transposed convolution. Since the operation is essentially the same, and only the (shape of the) kernel is changed between the two operations, the expected value and variance are the same as in the convolution operation.

3. Normally Distributed Data Assumption

In order to calculate the expected value and variance for the case of ReLU activation function, we made the assumption that all input data to the activation function is normally distributed. This assumption is supported by the following empirical evidence.

We conducted a series of experiments in order to evaluate the distribution of data after some ANN operations. This data is provided as input to a non-linear activation function. As discussed in Sec. 2, all the above operations involve linear combinations of different operand types. More specifically, the operands are of type: Uniform, Normal, and Scalar. Therefore, we only need to show that summations of different operand types result in Normal-like distributions.

Table 1 shows the histograms of sums for different combinations of X and Y operands for different quantities of

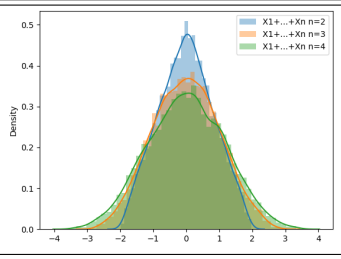
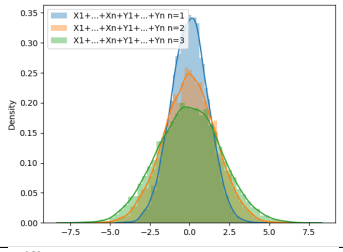
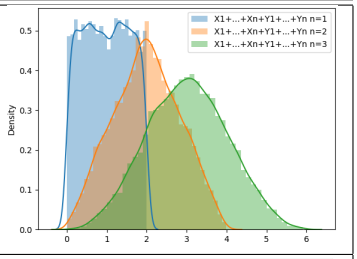
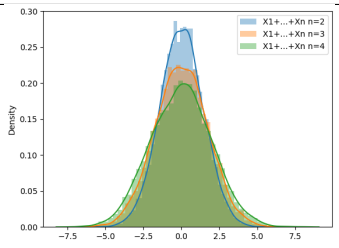
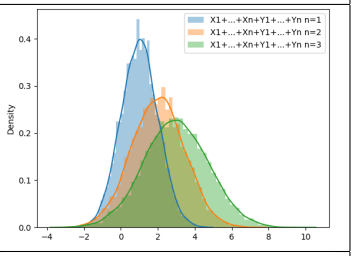
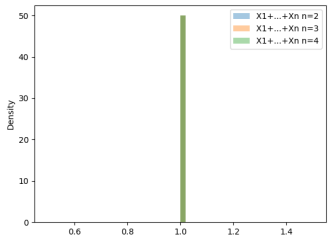
Operand	Y = Uniform	Y = Normal	Y = Scalar
X = Uniform			
X = Normal	symmetric plot		
X = Scalar	symmetric plot	symmetric plot	

Table 1: Histograms of summations for combination of different operands, where n is the number of operands.

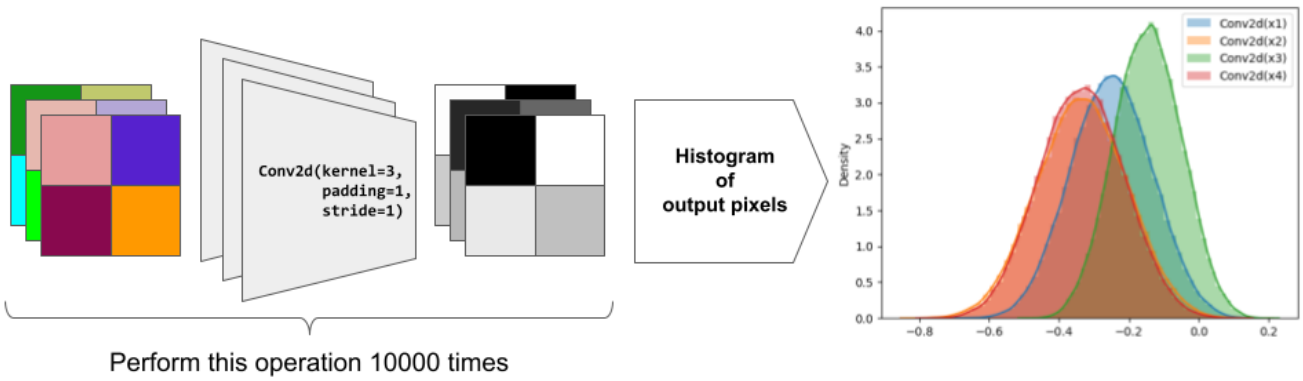


Figure 1: The histogram of 4 output pixels after a convolution operation. The histogram was obtained by repeating the same process for $10,000$ times. The input pixels were sampled each time from $\mathcal{U}(0,1)$.

operands. The histograms show the density of $10,000$ sampled values, where each sample is the sum of n operands. For every type of operand, if an RV X is of type Uniform then $X \sim \mathcal{U}(-1,1)$, if X is of type Normal then $X \sim \mathcal{N}(0,1)$, if X is of type Scalar then $X \sim \mathcal{N}(1,0)$, therefore for each sample each operand is drawn/sampled from its respective distribution. As we can observe, in most cases the resulting histograms take a Gaussian form even for a few operands. If all operands are of scalar type, we notice that, as expected, the output distribution is a Dirac delta function. We can describe this distribution in a Gaussian-like form as $\mathcal{N}(m,0)$ where m is the output scalar. In the

case of uniform plus scalar operands, we can observe that the output distribution is similar to an Irwin-Hall distribution, which is the case of uniform plus uniform. Therefore, for a large number of operands (specifically $n > 2$) the output yields a Gaussian-like distribution (third "green" histograms of Table 1).

More empirical results are reported in Fig. 1. For an input noise image where (the value of) each pixel is sampled from $\mathcal{U}(0, 1)$, we show the histogram of the output pixel values (4 pixels) after a convolution operation, repeated for $10,000$ times. Even though the input is uniformly distributed, the output pixel values follow a Gaussian-like distribution.

Figure 2 depicts the same operation as above but performed in a single RV convolution. For an equivalent input RV matrix, we show the plot of 4 Normal distributions created after a single RV convolution operation. We can observe that the distributions created are nearly identical to the histograms of Fig. 1.

4. More Qualitative Results

4.1. Image reconstructions

In the following Figs. 3 to 8 we present some more image reconstruction results for different architectures and datasets.

4.2. Image generations

In Figs. 9,10 we present some more image generation results for different architectures and datasets.

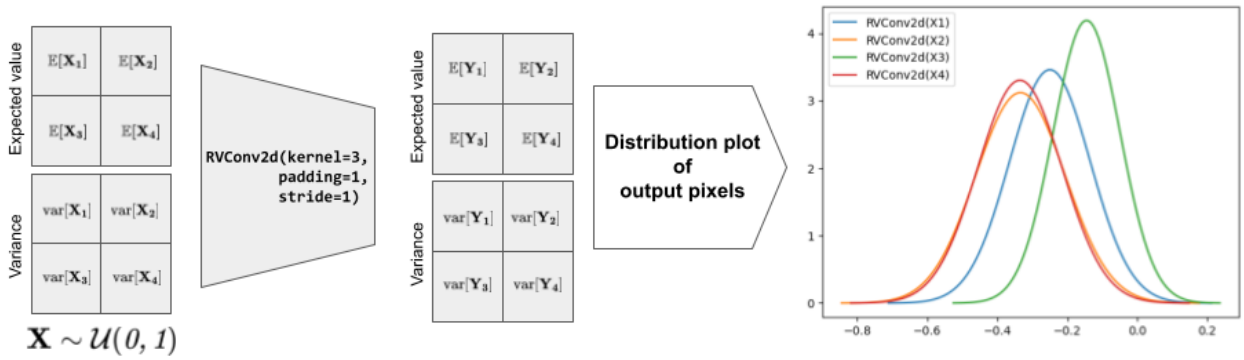


Figure 2: The plot of 4 Normal distributions created from the 4 output pixels after a single RV-convolution operation. The means and variances of those 4 Normal distributions are the respective means and variances of the output pixels'. The input values were RVs with expected value and variance $\mathbb{E}[\mathcal{U}(0, 1)]$ and $\text{var}[\mathcal{U}(0, 1)]$ respectively.

Figure 3: Reconstructions of CelebA real images (first row) using original VAE architecture (second row) and the proposed RV-VAE version (third row).

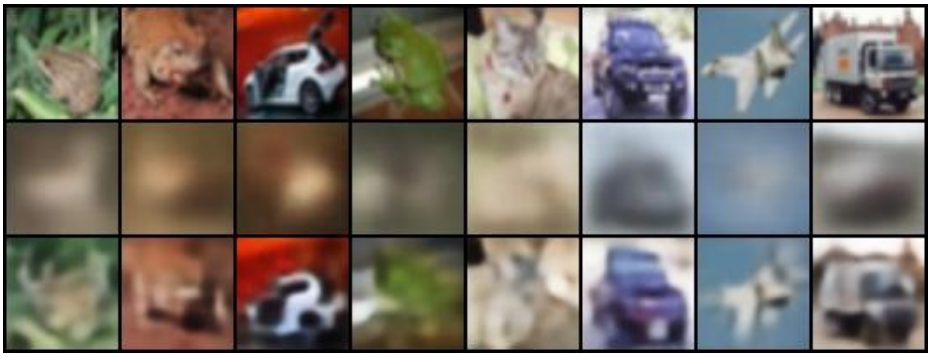


Figure 4: Reconstructions of CIFAR-10 real images (first row) using original VAE architecture (second row) and the proposed RV-VAE version (third row).

Figure 5: Reconstructions of CelebA real images (first row) using original β-TCVAE architecture (second row) and the corresponding proposed RV-aware version (RV-β-TCVAE version, third row).

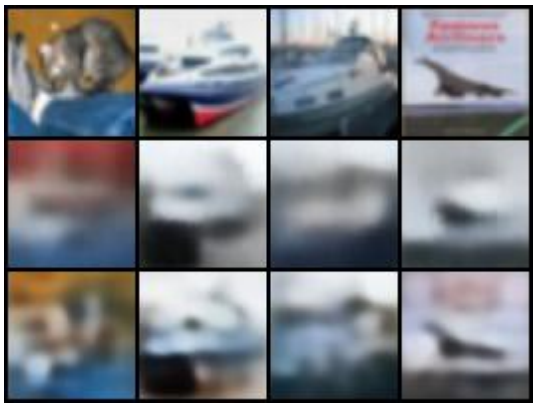


Figure 6: Reconstructions of CIFAR-10 real images (first row) using original β-TCVAE architecture (second row) and the corresponding proposed RV-aware version (RV-β-TCVAE version, third row).

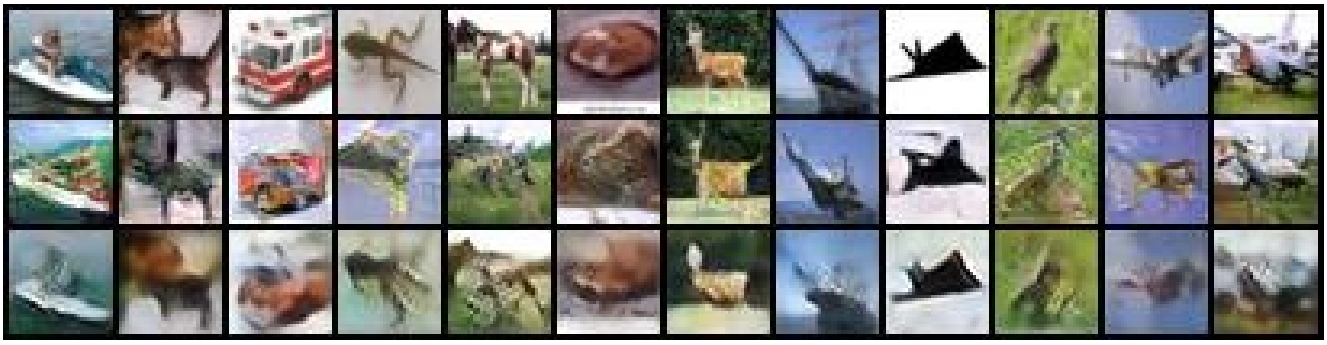


Figure 7: Reconstructions of CIFAR-10 real images (first row) using original Soft-Intro-VAE architecture (second row) and the corresponding proposed RV-aware version (RV-Soft-Intro-VAE version, third row).

Figure 8: Reconstructions of CelebA-HQ real images (first rows) using original Soft-Intro-VAE architecture (second rows) and the corresponding proposed RV-aware version (RV-Soft-Intro-VAE version, third rows).

(a) VAE (b) RV-VAE

Figure 9: Image generations on CelebA with (a) VAE and (b) the corresponding proposed RV-aware version (RV-VAE).