**ICP7**

**BIG DATA ANALYTICS AND APPLICATIONS(CS5542)**

*Sravani Niharika Garapati (16328843)*

*Nikhila Chirumamilla (16323180)*

**QUESTION:**

Using the given dataset and model provided, must add two layers for encoder path and their corresponding two layers in the decoder path.

**What I have learned from this ICP:**

In this ICP, professor asked us to implement variational auto encoders by adding two more layers in both encoder and decoder path. The things which I have learned from this ICP are listed below

* I have learned many functions that were very useful in implementing the variational auto encoder such as encoder and decoder implementation functions.
* I have known in detail about the taken data set and to perform analysis and performing manipulations on that data set in a thorough manner.
* **MNIST Dataset:**

The MNIST database contains 60,000 training images and 10,000 testing images. The **MNIST** (Modified National Institute of Standards and Technology) is a large database of handwritten digits that is commonly used for training various image processing systems. It is a dataset of 60,000 small square 28×28-pixel grayscale images of handwritten single digits between 0 and 9.

* I have learned many functions that were very useful in implementing the model and finding losses.
* I have known which model and which layers to be used in the situation.
* I have come to know that increasing number of epochs while fitting the model would give more accuracy and reduce the loss.
* Also, I have known changing number of strides also makes the model work effective and reduce loss and would generate the output in an effective manner.
* I have known about encoder and decoder functionalities and their implementations.

**ICP Description about the Task:**

In this ICP, we implemented variational auto encoders by adding two more layers in both encoder and decoder path. This is done in the following steps

* As first step, I have imported all the necessary libraries.
* Then, download the dataset and read the contents in the dataset as the second foremost step.
* Reduced the data size here (training set is reduced from 60000 to 10000 images, and test set is reduced from 10000 to 1000 images) because the original data size was causing out of memory errors and requiring extra GPU.
* Then, I have set up the configuration parameters such as image wight, height, batch size, number of epochs, validation split, verbosity, latest dimensions and number of channels for data and model
* Then, we reshape the data so that it takes the shape (X, 28, 28, 1), where X is the number of samples in either the training or testing dataset. We also set (28, 28, 1) as input shape and then we parsed the numbers as floats, which presumably speeds up the training process, and normalized it, which the neural network appreciates
* Then defined the auto encoder for model 1

**Model-1(Encoder Definition)**

* The first layer is the **Input laye**r. It accepts data with input\_shape = (28, 28, 1) and is named encoder\_input
* Next up is a two-dimensional convolutional layer, or **Conv2D** in Keras terms with 8 filters deploying 3\*3 kernel which it convolves over the input.
* Used **Batch Normalization** which ensures that the outputs of the Conv2D layer that are input to the next Conv2D layer have a steady mean and variance
* Again, a Conv2D layer with 16 filters and for the rest is equal to the first Conv2D layer.
* BatchNormalization layer once more.
* Next up, a **Flatten layer**, it only serves to flatten the multidimensional data from the convolutional layers into one-dimensional shape. This has to be done because the densely connected layers that we use next require data to have this shape.
* The next layer is a **Dense** layer with 20 output neurons. It’s the autoencoder bottleneck we’ve been talking about.
* BatchNormalization once more.
* And at last mu and sigma layers.
* The next step is to retrieve the shape of the final Conv2D output which we will need it when defining the layers of our decoder.
* Then, defined sampling with reparameterization trick.
* Then, we used that reparameterization trick to ensure correct gradient for mu and sigma.
* We instantiatedthe Encoder
* Then defined the auto decoder for model 1

**Model-1(Decoder Definition)**

* Decoder also starts with an Input layer, the **decoder\_input** layer. It takes input with the shape (latent\_dim)
* Then, we upsample the point in latent space with **Conv2DTranspose** layers, in exactly the opposite symmetrical order as with we downsampled with our encoder.
* Added a **Dense layer** which has conv\_shape [1] \* conv\_shape [2] \* conv\_shape [3] output and converts the latent space into many outputs.
* Next use a Reshape layer to convert the output of the **Dense layer** into the output shape of the last convolutional layer.
* We then used Conv2DTranspose and **BatchNormalization** in the exact opposite order as with our encoder to upsample our data into 28 x 28 pixels.
* We therefore added a final Conv2DTranspose layer which does nothing to the width and height of the data but ensures that the number of filters learns equals num\_channels.
* We instantiatedthe Decoder.
* We instantiatedthe VAE
* Then we compiled the model
* **Optimizer Used:** Adam
* **Loss function used:** binary\_crossentropy
* **Loss Observed:** 0.1893
* And then wrote functions to visualize the results and plotted the results.
* Then defined the auto encoder for model 2

**Model-2(Encoder Definition)**

* Encoder definition for Model 2 is same as Model 1 but have added two extra layers in models 2
* Added Conv2d layers with filters 32 and 64 and also changed the number of strides from 2 to 1 in order to reduce the loss
* With 2 Conv2D layers, we have also added two batch normalization () layers which ensures that the outputs of the Conv2D layer that are input to the next Conv2D layer have a steady mean and variance
* These changes are done to reduce the loss and to produce the output in an effective manner.
* We instantiatedthe Encoder.
* Then defined the auto Decoder for model 2

**Model-2(Decoder Definition)**

* Decoder definition for Model 2 is same as Model 1 but have added two extra layers in models 2
* Added Conv2dTranspose layers with filters 32 and 64 and changed the number of strides from 2 to 1 in order to reduce the loss
* With 2 Conv2DTranspose layers, we have also added two batchnormalization () layers which ensures that the outputs of the Conv2DTranspose layers that are input to the next Conv2DTranspose layer have a steady mean and variance
* These changes are done to reduce the loss and to produce the output in an effective manner.
* We instantiatedthe Decoder.
* We instantiatedthe VAE
* Then we compiled the model
* **Optimizer Used:** Adam
* **Loss function used:** binary\_crossentropy
* **Number of Epochs:** 100
* **Loss Observed:** 0.1688
* And then wrote functions to visualize the results and plotted the results.

**Challenges Faced in this ICP:**

The challenges which we faced while doing this ICP are listed below:

* It has become very difficult for us to identify which other 2 layers to be added to the encoder and decoder to reduce the loss.
* Also, it has been very difficult for me to understand the functions that are involved in plotting and visualizing results.

**Screenshots of the execution:**

*Importing required Libraries*

*Graphical user interface, text, application, email

Description automatically generated*

*Loading the MNIST Dataset*

*Graphical user interface, text, application, email

Description automatically generated*

*Depicting the training and testing data and their shapes*

*Table

Description automatically generated*

*Reducing the Data size*

*Graphical user interface, application, Word

Description automatically generated*

*Printing the training and test data sizes after reducing the size*

*Text

Description automatically generated*

*Printing first 20 images of the dataset*

*Graphical user interface

Description automatically generated with low confidence*

*Model Configuration*

*Graphical user interface, text

Description automatically generated*

*Reshaping and normalizing the data*

*Text, application

Description automatically generated*

*Auto Encoder Definition*

*Text

Description automatically generated with medium confidence*

*Layers Description in Encoder*

*Text

Description automatically generated*

*Definition of the encoder*

*Text

Description automatically generated*

*Retrieving the shape of the final Conv2D output*

*Graphical user interface, text, website

Description automatically generated*

*Defining Sampling with reparameterization*

*Graphical user interface, text, application, email

Description automatically generated*

*Using Lambda to ensure correct gradients*

*Text

Description automatically generated*

*Encoder Instantiation*

*Graphical user interface

Description automatically generated with medium confidence*

*Decoder Definition and Layers Description*

*Text

Description automatically generated*

*Decoder Definition*

**Text

Description automatically generated**

*Decoder Instantiation*

*Table

Description automatically generated with medium confidence*

*Instantiating VAE*

*Text

Description automatically generated*

*Compiling the Model*

*Table

Description automatically generated*

*Visualization of Results*

*Text

Description automatically generated*

*Function for plotting*

*Text

Description automatically generated*

*Text

Description automatically generated*

*Chart, scatter chart

Description automatically generated*

*Plot*

*A picture containing calendar

Description automatically generated*

*Model 2(Encoder Definition)*

*Text

Description automatically generated with low confidence*

*Sampling and reparameterization trick*

*Graphical user interface, text, application

Description automatically generated*

*Instantiation of Encoder*

*Text

Description automatically generated with medium confidence*

*Decoder Definition*

*Text

Description automatically generated*

*Instantiation of Decoder*

*Table

Description automatically generated*

*Instantiating VAE*

*Text

Description automatically generated with low confidence*

*Compiling & Training*

*Text, table

Description automatically generated with medium confidence*

*Plotting the results*

*Graphical user interface, application, Word

Description automatically generated*

*Chart, scatter chart

Description automatically generated*

*Plot*

*A picture containing text, electronics, display, computer

Description automatically generated*

**Video Link**

https://youtu.be/DUW0Xkq\_QyI