Control-VAE On MNIST Dataset

- **Intro**
- **NN structure**
- **Models**
- **Training & results**
- **Reconstruction Quality**
- **Conclusion**
- **Latent Spaces**
- **Code explanation**

- ***Intro***

  We used MNIST dataset to conduct our experiment, to examine the improvement of Control-VAE on the reconstruction loss and ELBO values, we trained two models with the same NN structure and differs with the beta value for *200 epochs* each, *Random_seed= 0*.

- ***NN structure***
  - Layers- Linear layers with ReLU activation function:

Encoder	**Decoder**
Input 784 Linear. ReLU	Input 10 Linear
400 Linear. ReLU	50 Linear. ReLU
50 Linear. ReLU	400 Linear. ReLU
2 Linear. ReLU	Output 784 Linear. Sigmoid
Output 2X2 Gaussian Latent	

  - Loss Function- the loss function used is Cross-Entropy loss
  - Optimization Algorithm : Adam optimizer with *0.0001 learning rate*

- ***Models***

  1- ***Beta-VAE*** has the VAE structure introduced above with *beta= 1*.

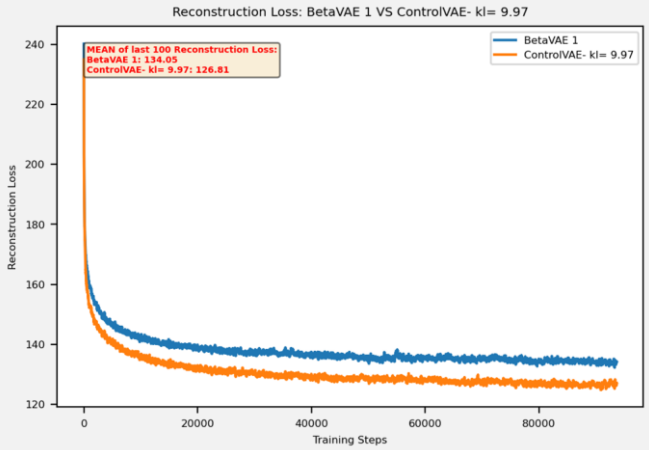
  2- ***Control-VAE*** has the VAE structure introduced above with the PI-Control algorithm that adjusts the beta value according to the desired KL Divergence input.

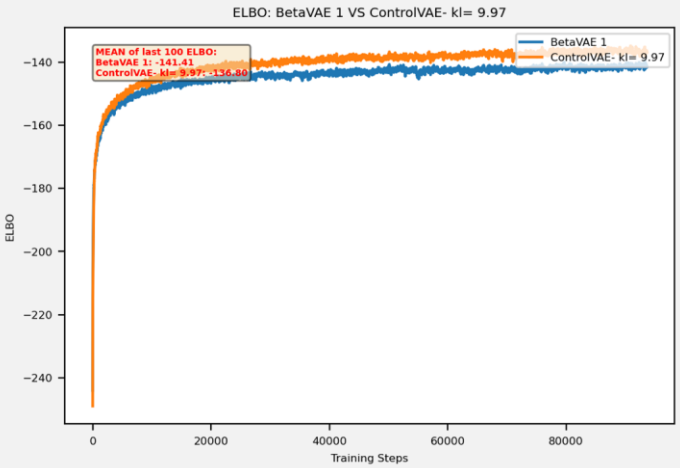
  in our case *desired-kl= 9.97* which equals to the optimal value that maximizes the ELPO and minimizes the Reconstruction Loss compared to the Beta-VAE KL-Divergence Value.

  PI-Control Function Parameters:

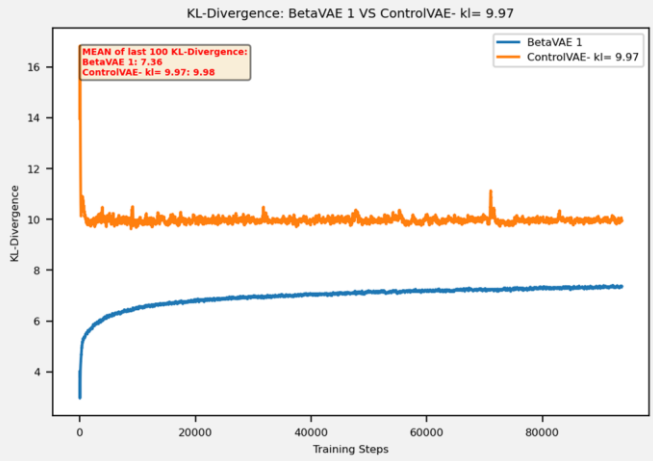
$K_p = 0.01$	$K_i = 0.0001$	$KL = 9.97$	$\beta_{min} = 0.01$	$\beta_{max} = 10$

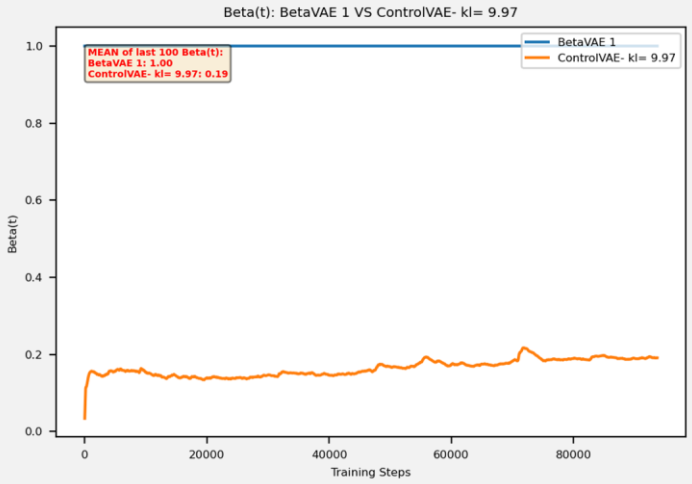
- ***Training & Results***
  The graphs below show results of training both β-VAE($\beta = 1$) and Control-VAE($KL = 9.97$) models for 200 epochs, the X axis shows for every step (batch) the mean values of the last N batches data along the epochs, random seed = 0.





- Control-VAE gets small RL's values faster than Beta-VAE at training time
- Control-VAE gets higher ELBO values than Beta-VAE, which indicates that the Control-VAE model does better on both Disentanglement of the latent variables and reconstructing data from those variables





- Beta-VAE: KL-Divergence value stays the same since the beta does not change
- Control-VAE: we see how the PI-Control algorithm changes the KL-Divergence value; at the first training steps the algorithm focuses on minimizing the RL error (learning the to reconstruct well from the latent space), at late training steps the beta value goes higher which allows the model to both minimize the reconstruction loss and the KL-Divergence.

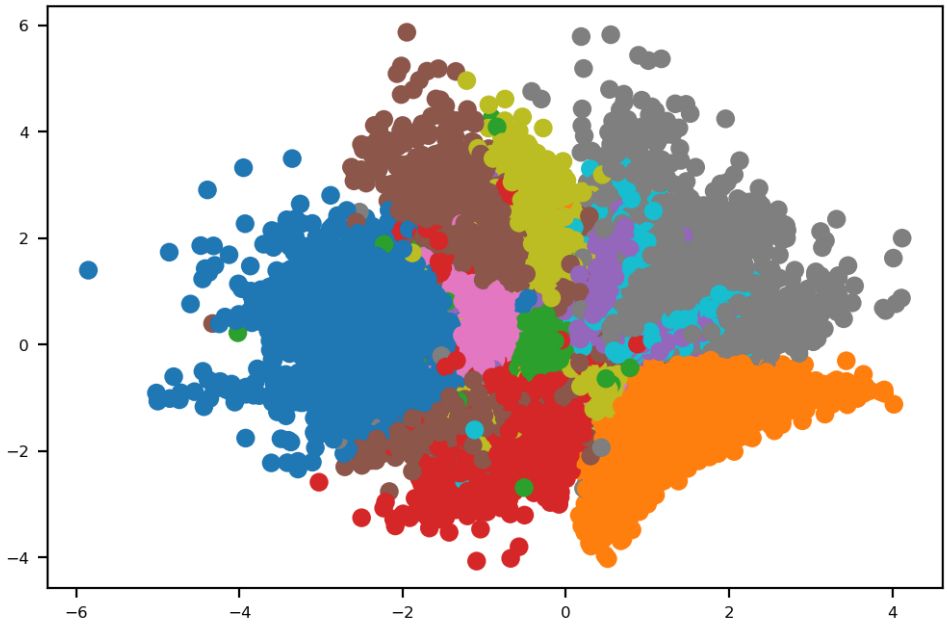
- Beta-VAE: Beta value stays the same at each training step
- Control-VAE: Beta value changes as the PI-Control algorithm measures the new beta according to the sampled KL input

- ***Reconstruction Quality***
  Ten original images of each digit are picked randomly from the MNIST Dataset, and fed to both networks, the results are shown below (Random seed =10):

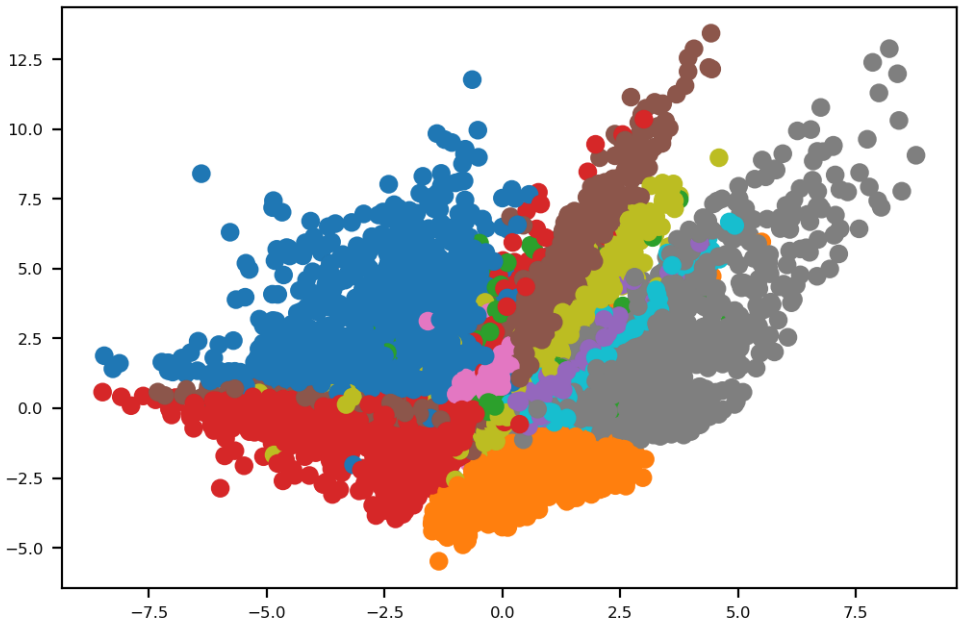
Original Images	
β-VAE($\beta = 1$)	
Control-VAE($KL = 9.97$)	

  - Notice that the Control-VAE achieved better reconstruction quality on numbers: 3, 8 and 2. The number 6 was challenging for both models reconstructing 5 in place of 6, yet Control-VAE seems to have a better prediction of the input picture.

- ***Latent Spaces***
  - *Beta-VAE Latent Space:*



  - *Control-VAE Latent Space:*



- Notice that the Control VAE segments the latent space better which indicates for better ***disentanglement*** of the latent variables as expected

- **Conclusion**
  The PI-Control algorithm for changing the value of β dynamically in training time on MNIST dataset gains the model an advantage for getting to better reconstruction quality and higher ELBO values. It shows that getting to the optimal KL-Divergence value through adjusting the value of β gains the model a systematic way to turn the focus of the training objective function.

- ***Code Explanation***
  Submission Contains 5 python files.
  - *Models_def.py* : a python file with models' definitions classes and functions
  - *train_models.ipynb* : a notebook file to define an instance of the model and train it
  - *plot_results.ipynb* : a notebook file to load trained models and plot results
  - *Model-BetaVAE 1.py* : trained Beta-VAE with Beta value of 1
  - *Model-ControlVAE kl= 9.97.py* : trained Control-VAE with desired KL value of 9.97

  To reconduct the experiment simply download files and run ***train_models.ipynb***