Learning Interpretable Representation for Fingerprinting Problem

Guillem Reus Muns and Somanshu Singh

1 Background and problem motivation

1.1 Introduction

The IoT (Internet of Things) relies on each networked device to honestly present a digital ID (identification). As Fig. 1 illustrates, digital IDs (in software and firmware) can be cloned, so IoT security would benefit if it were supplemented by a unique physical ID intrinsic to the device. RF transmitters are naturally imperfect devices due to the tolerances in manufacturing of the analog electronics.

A radiofrequency fingerprint (RFFP) for each transmitting device can be measured. The communications security literature has many examples of the RFFP concept. The typical research centers around a few nominally identical transmitters of a single protocol type and studies their transmit signals in order to hand-engineer features that discriminate the individual transmitters. While successful as proof of concept demonstrations, hand-engineered-feature discriminators have not been shown to support the number of devices expected in IoT populations. Similarly, hand-engineering approaches which focus on only a single type of RF system (e.g. WiFi) will not handle the diversity of wireless protocols emerging in the IoT. To sum up, we suggest using the impairments intrinsic in the RF devices in order to find a unique fingerprint that identifies each one.

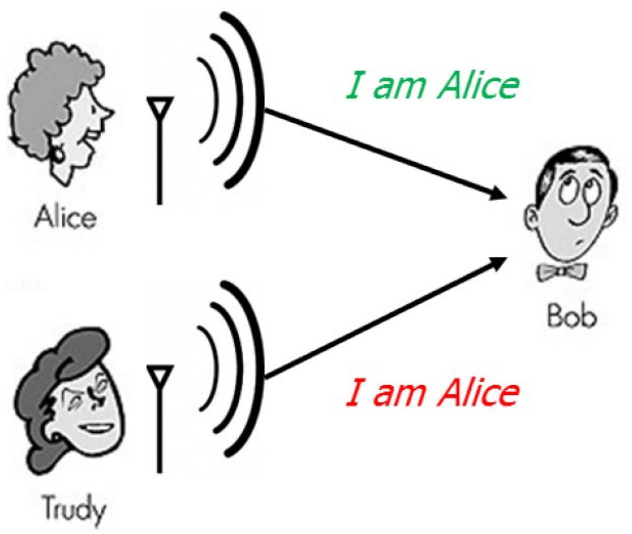

Figure 1: Bob would identify Trudy is not really Alice.

1.2 Hardware impairments

A block diagram of transceiver pair is shown in Fig. 2, with the main causes of shifts in the received complex valued I/Q samples highlighted. We first study the effect of the hardware-induced causes of I/Q deviation from the ideal values. Our preliminary studies on understanding how I/Q samples are impacted by the transmitter are motivated by our desire to **exploit a key input feature: the displacement of the received I/Q sample from the canonical location of its closest constellation point**. Using the MATLAB Communication System Toolbox, we design a simulation model of a typical wireless communications processing chain, and then modify the ideal operational blocks to introduce RF impairments, typically seen in actual hardware implementations. This allows us to individually study the I/Q imbalance, phase noise, carrier frequency and phase offset, and nonlinearity of power amplifier, in isolation of each other:

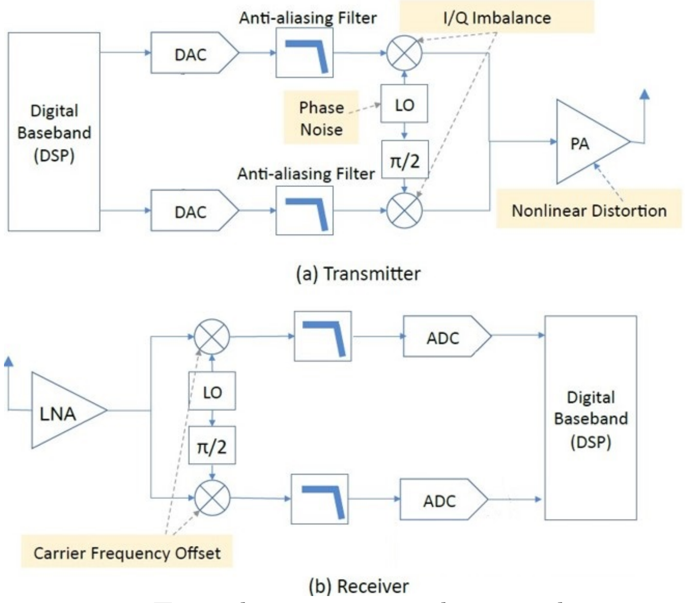

Figure 2: Typical transceiver chain with various sources of RF impairments.

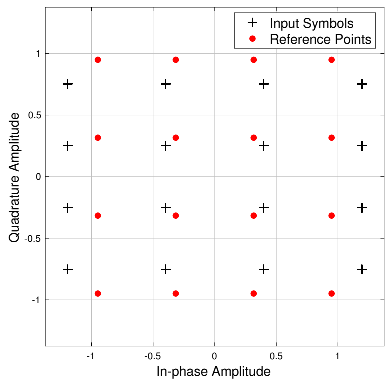

Figure 3: Amplitude Imbalance

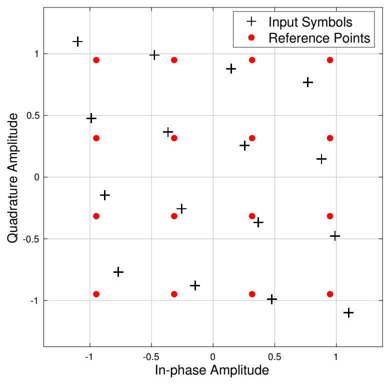

Figure 4: Phase Imbalance

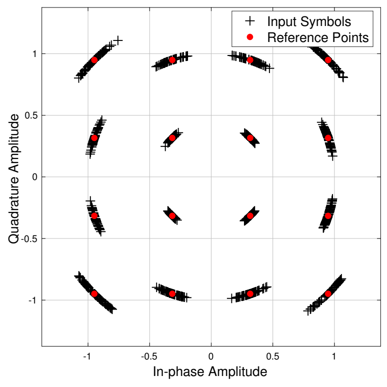

Figure 5: Phase Noise

•**I/Q imbalance:**Quadrature mixers that convert baseband to RF and vice versa are often impaired by gain and phase mismatches between the parallel sections of the RF chain dealing with the in-phase (I) and quadrature (Q) signal paths. The analog gain is never the same for each signal path and the difference between their amplitude causes amplitude imbalance. In addition, the delay is never exactly 90°, which causes phase imbalance. Fig. 3 and 4 illustrate the effect of amplitude imbalance and phase imbalance on a 16-QAM constellation. In practice, I/Q amplitude imbalance is expressed in the range [-5, 5] dB, whereas phase imbalance in the range [-30, 30] degrees. Knowing these ranges can help create a tractable search space for intentional modification of the baseband waveform.

•**Phase Noise**: The up-conversion of a baseband signal to a carrier frequency f_c is performed at the transmitter by mixing the baseband signal with the carrier signal. Instead of generating a pure

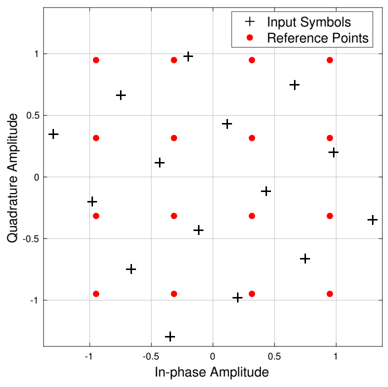

Figure 6: Phase Offset

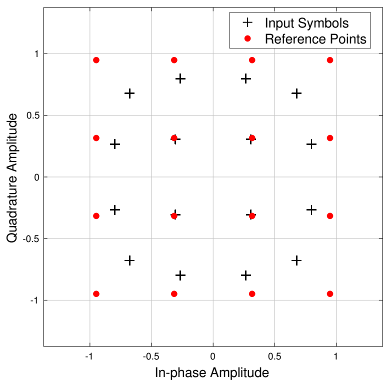

Figure 7: AM/AM Distortion

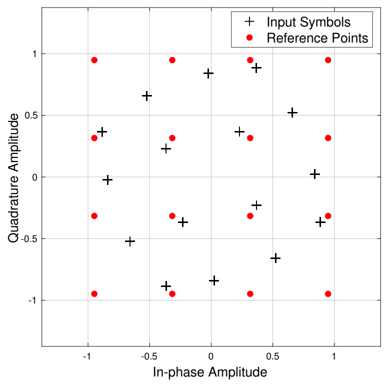

Figure 8: AM/PM Distortion

tone at frequency f_c, i.e., $e^{j2\pi f_c t}$, the generated tone is actually $e^{j2\pi f_c t+\phi(t)}$, where $\phi(t)$ is a random phase noise. The phase noise introduces a rotational jitter as shown in Fig. 5. Phase noise is expressed in units of dBc/Hz, which represents the noise power relative to the carrier contained in a 1 Hz bandwidth centered at a certain offset from the carrier. Typical values of phase noise level is in the range $[-100,\ -48]$ dBc/Hz, with frequency offset in the range $[20,\ 200]$ Hz.
• **Carrier Frequency and Phase offset**: The performance of crystal oscillators used to generate the carrier frequency is specified with a certain accuracy in parts per million (ppm). The difference in transmitter and receiver carrier frequencies is referred to as carrier frequency offset (CFO). Due to CFO, the received signal spectrum shifts by a frequency offset:

$$y(t) = x(t)e^{j2\pi(f_{Tx}-f_{Rx})t} = x(t)e^{j2\pi\Delta_{CFO}t} \tag{1}$$

where Δ_{CFO} is the shift introduced by CFO between transceiver.
•**Power amplifier distortions**: Power amplifier (PA) non-linearities mainly appear when the amplifier is operated in its non-linear region, i.e., close to its maximum output power, where significant compression of the output signal occurs. The distortions of the power amplifiers (PA) are generally modeled using AM/AM (amplitude to amplitude) and AM/PM (amplitude to phase) curves. The AM/AM causes amplitude distortion whereas AM/PM introduces phase shift. As shown in Fig. 7, the corner points of the constellation have moved toward the origin due to amplifier gain compression. The constellation has rotated due to the AM/PM conversion in Fig. 8. The nonlinearity of amplifier is modeled using Cubic Polynomial and Hyperbolic Tangent methods using Third-order input intercept point (IIP3) parameter. IIP3 expressed in (dBm) represents a scalar specifying the third order intercept.

All these impairment present on each device should be useful in order to identify them and neural networks are a strong candidate to consider in order to perform that task.

1.3 Motivation on using disentangled representations

As mentioned on the previous sections, we want to use those physical properties of the devices in order to identify them. However, the intended receiver who should perform the classification, will receive each an every signal effected by the channel as well. In a very simplistic manner, the channel can be understood as the difference any transmitted signal (after the transmitter "applied" its own impairments) and the received one. Therefore, we have another player with a very important role in the game of fingerprinting. It is important to understand that any channel effect should not be considered at all. Otherwise, the classification would be correlated with the environment and would not generalize in different scenarios. Also, it may happen that two signals coming from different devices may be effected by the same channel. In conclusion, we want to use NNs to train a classifier but at the same time its mandatory to be able to know and understand what is our network learning. Therefore, disentangled variational autoencoders are our chosen candidate for achieving such goal.

2 Disentangled VAEs

Learning an interpretable factorized representation of the independent samples from a wireless transmitter can be done using many methods that have been recently proposed.We use two of those methods, β-VAE and β-TCVAE, to achieve our goal.

2.1 β-VAE

β-VAE is a simple modification of a normal Variational Autoencoder. It introduces a adjustable hyper-parameter β that balances latent channel capacity of the network and independence constraints with the reconstruction quality of the network. ($\beta > 1$) corresponds to network performing disentanglement, while for ($\beta = 0$), the network behaves as a normal Variational Autoencoder.

2.1.1 β-VAE Derivation

A suitable objective is to maximize the marginal (log-)likelihood of the observed data $\mathbf{x}$ in expectation over the whole distribution of latent factors $\mathbf{z}$:

$$\max_{\theta} \mathbb{E}_{p_\theta(z)} \left[p_\theta(x|z) \right] \tag{2}$$

For a given observation x, the inferred posterior configurations of the latent factors $\mathbf{z}$ is described by a probability distribution $q_\phi(x|z)$. In order to encourage the disentangling property in the inferred $q_\phi(x|z)$, a constraint is introduced over it by trying to match it to a prior $p(z)$ that can both control the capacity of the latent information bottleneck, and embodies the desiderata of statistical independence mentioned above. This can be achieved if the prior is set to be an isotropic unit Gaussian ($p(\text{z}) = \mathcal{N}(0, I)$), hence arriving at the constrained optimization problem in Eq. 3, where ϵ specifies the strength of the applied constraint.

$$\max_{\phi,\theta} \mathbb{E}_{x\sim D} \left[\mathbb{E}_{q_\phi(z|x)} \left[log\ p_\theta(x|z) \right] \right] \qquad \text{subject to } D_{KL} \left(q_\phi(z|x) | p(z) \right) < \epsilon \tag{3}$$

Re-writing Eq. 3 as a Lagrangian under the KKT conditions, we obtain:

$$\mathcal{F} \left(\theta, \phi, \beta; x, z \right) = \mathbb{E}_{q_\phi(z|x)} \left[log\ p_\theta(x|z) \right] - \beta \left(D_{KL} \left(q_\phi(z|x) \| p(z) \right) - \epsilon \right) \tag{4}$$

where the KKT multiplier β is the regularization coefficient that constrains the capacity of the latent information channel z and puts implicit independence pressure on the learned posterior due to the isotropic nature of the Gaussian prior p(z) Since β, $\epsilon \geq 0$ according to the complementary slackness KKT condition, Eq. 4 can be re-written to arrive at the β-VAE formulation.

$$\mathcal{F}(\theta, \phi, \beta; x, z) \geq \mathcal{L}(\theta, \phi, ; x, z, \beta) = \mathbb{E}_{q_\phi(z|x)}\left[log\ p_\theta(x|z)\right] - \beta\left(D_{KL}\left(q_\phi(z|x) \| p(z)\right)\right) \tag{5}$$

Varying β, changes the degree of applied learning pressure during training, thus encouraging different learnt representations. β-VAE where $\beta = 1$ corresponds to the original VAE formulation.

$\beta > 1$ puts a stronger constraint on the latent bottleneck than in the original VAE formulation. These constraints limit the capacity of z, which, combined with the pressure to maximize the log likelihood of the training data x under the model, encourages the model to learn the most efficient representation of the data.

The extra pressures coming from high β values, however, may create a trade-off between reconstruction fidelity and the quality of disentanglement within the learned latent representations. Disentangled representations emerge when the right balance is found between information preservation (reconstruction cost as regularization) and latent channel capacity restriction ($\beta > 1$). The latter can lead to poorer reconstructions due to the loss of high frequency details when passing through a constrained latent bottleneck.

2.2 β-TCVAE

β-TCVAE is a refinement of the -VAE for learning disentangled representations without supervision.

β-TCVAE decompose the mean of D_{KL} term into term given as,

$$\mathbb{E}_{p(n)}\left[D_{KL}\left(q(z|n) \| p(z)\right)\right] = \underbrace{D_{KL}\left(q(z|x_n) \| q(z)p(n)\right)}_{\text{Index-Code MI}} + \underbrace{D_{KL}\left(q(z) \| \prod_j q(z_j)\right)}_{\text{Total Correlation}} + \underbrace{\sum_j D_{KL}\left(q(z_j) \| p(z_j)\right)}_{\text{Dimension-wise KL}} \tag{6}$$

A low total correlation is the main reason β-VAE achieves empirical success in learning disentangled representations. However, the β-VAE also encourages the model to discard information from the latents since it penalizes the index-code mutual information.

The β-TCVAE objective is:

$$\begin{aligned}\mathcal{L}_{\beta-TC} := \frac{1}{N}\sum_{n=1}^{N}\left(\mathbb{E}_{q(z|n)}\left[log\ p(n|z)\right]\right) - \alpha I_q(z; n) \\ - \beta D_{KL}\left(q(z) \| \prod_j q(z_j)\right)\end{aligned} \tag{7}$$

where $\beta > 1$ and $\alpha = \gamma = 1$.

To obtain $q(z)$ and $q(z_j)$, training is done using *Mini-Batch Stratified Sampling(MSS)*

2.2.1 Mini-Batch Stratified Sampling(MSS)

During training, we sample a mini-batch of size M without replacement that does not contain n^*. We estimate the first term using n^* and M 1 other samples, and the second term using the M samples that are not n^*. One can also view this as sampling a mini-batch of size M+1 where n^* is one of the elements,and let $B_{M+1} \backslash n^* = \hat{B}_M = n1, ..., n_M$ be the elements that are not equal to n^* , then we can estimate the first expectation using $n^* n1, ..., n_M 1$ and the second expectation using $n1, ..., n_M$. This estimator can be written as:

$$q(z) \approx f(z, n^*, \hat{B}_M) = \frac{1}{N} q(z|n^*) + \frac{1}{M} \sum_{m=1}^{M-1} q(z|n_m) + \frac{N-M}{NM} q(z|n_M) \tag{8}$$

3 Results

We have chosen two different architectures for our project. In the first one, the encoder is formed by convolutional layers, however, in the bottleneck we attached a classifier (which takes as an input the means and variances, before sampling the z) and a decoder. The aim of this architecture was forcing the network to learn to classify and a generative model in parallel. The function were maximized in that case was the following:

$$\mathcal{L}(\theta, \phi, ; x, z, \beta) = \mathbb{E}_{q_\phi(z|x)} [log\ p_\theta(x|z)] - \beta (D_{KL} (q_\phi(z|x) \| p(z))) - \alpha \mathcal{E}_{loss} \tag{9}$$

where $\mathcal{E}_{loss}$ is the classifier loss. The second one is a VAE with convolutional layers. After the trained was done, we removed the decoder and plugged in a classifier with the learned features and trained that one as well. For both models we have used β-VAE and β-TCVAE. In the following plots, we want to show the trade off between accuracy, and the parameters α and β for model 1 in both β-VAE and β-TCVAE. As for model 2, we will only show the accuracy in terms of different β values. Those results were on MNIST dataset. Due to the long training time, we only had time to train model 2 for β-VAE once. We got results of about 81%. However, we really cannot give an interpretation to those results without being able to analyze the disentanglement and train for different parameters.

In Fig. 9 we can see how Alpha shows better classification results, which makes sense as with alpha we are weighting the classification error over the other terms in the function. Also, we can see how TC has given far better results in classification as well. Alpha's effect can also be obesrved in Fig. 10. The effect of Beta is the opposite, improves disentaglement but gives worse classification. In Fig. 11 we see how increasing β, as we have more disentanglement, the accuracy decreases. This may be because when we force disentanglement, if two variables are correlated, the network will only learn the most important one and drop the other. Finally, we just wanted to show a few numbers generated using our generative model.

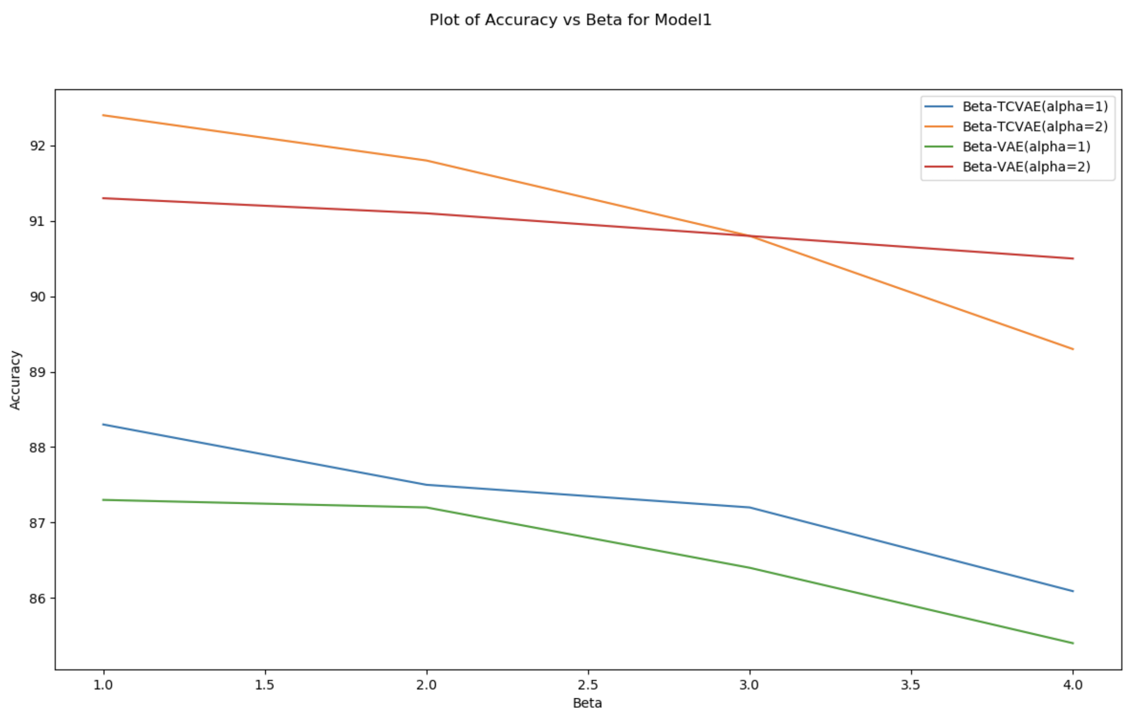

Figure 9: Effect of Beta

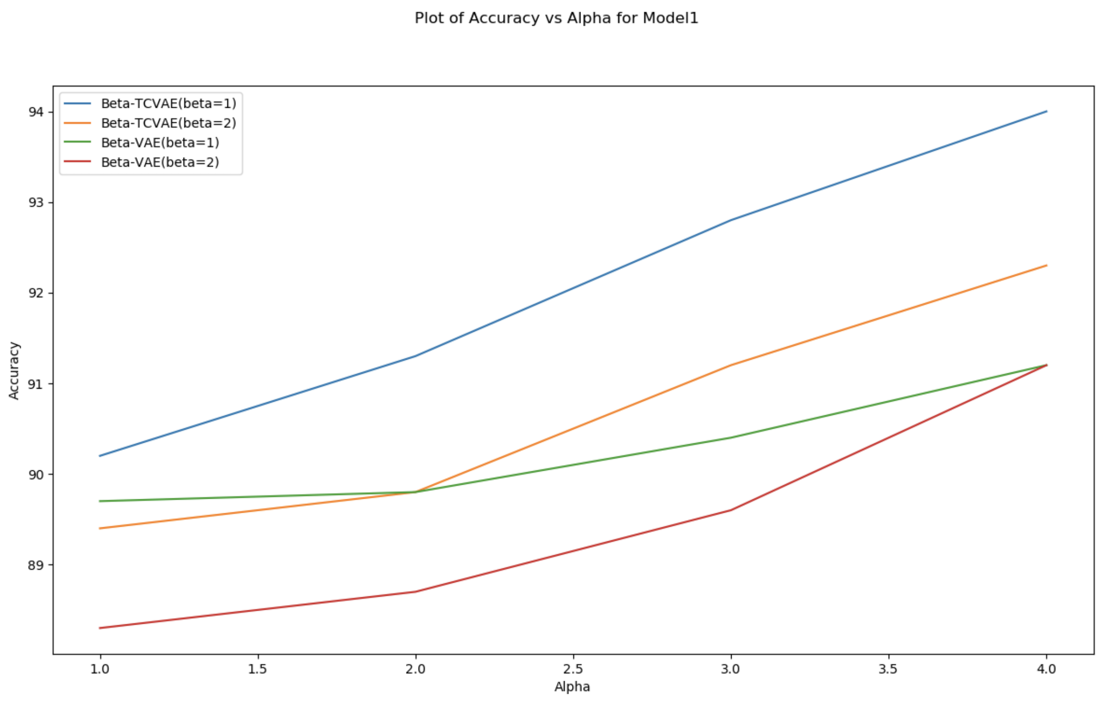

Figure 10: Effect of Alpha

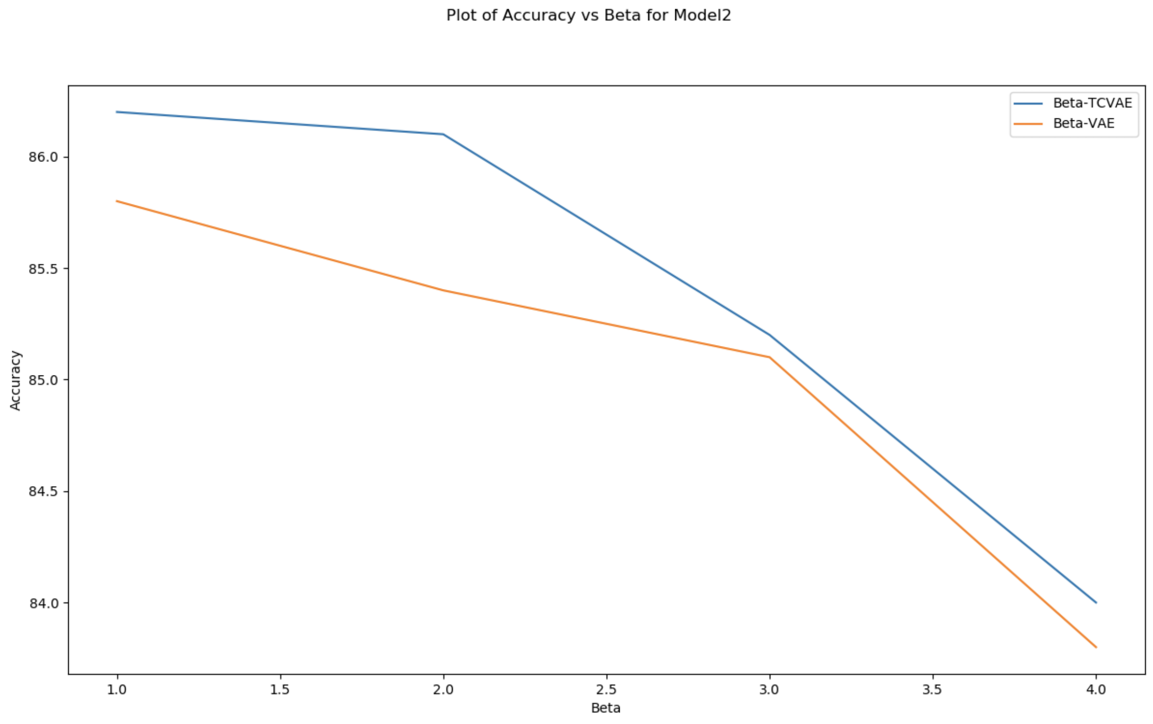

Figure 11: Effect of Beta

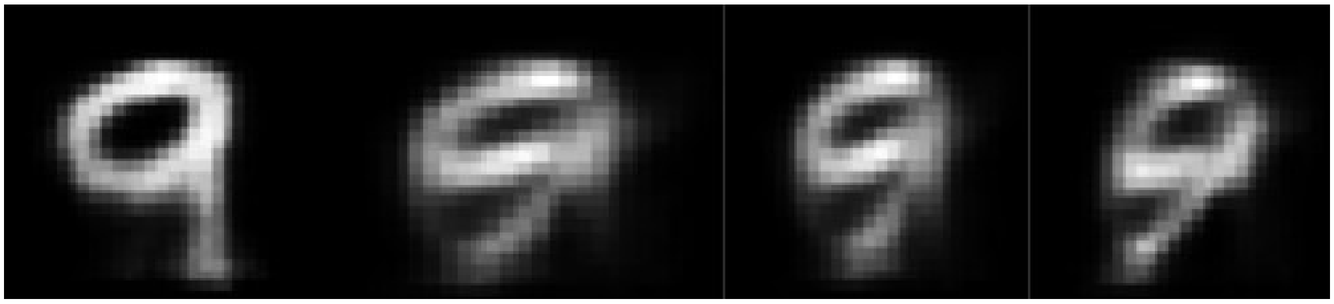

Figure 12: Samples from MNIST dataset, only sampled from one variable. Different tilting can be interfered in each image.