Project Report

An Expose on Kullback-Leibler Divergence

by

Faizan Shaikh Abdul Khalil Shaikh

for the course - UAI/500 Information Theory

Abstract

The writeup is an in-depth exploration of Kullback Leibler divergence (KLD), a widely used metric to measure two probability distributions. I will gently introduce the topic along with an example to familiarize ourselves with the terminologies. We will then dive into the mathematical aspects of KLD by formulating a theoretical derivation of KLD for two univariate gaussian distributions. Lastly, I will lay out a practical application of KLD as a learning metric for a deep learning architecture called Variational Autoencoders.

Introduction

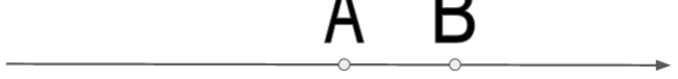
One of my excursions into the Deep Learning space, surveying unsupervised deep learning, brought me to the topic of KL Divergence. Although I have encountered it once or twice, I have never explored the topic thoroughly. I figured this would be the opportunity to do so, as I can make this my class project for Information Theory (two for one deal :))

We've established that we are going to examine the term KL Divergence. But what exactly is KL Divergence? A textbook definition is that

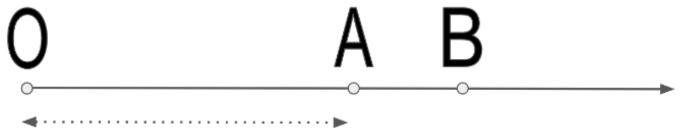
"Kullback-Leibler Divergence is a measure of comparison between two probability distributions" [1]

Let's dissect this definition.

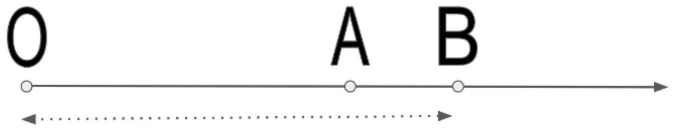
Say you have two points A and B on a line. How do you compare them?



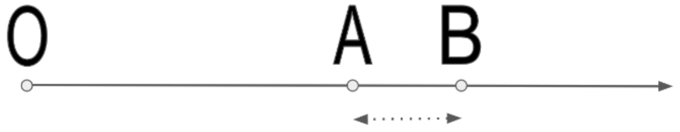
One way could be that we define an arbitrary point as origin O. Now we can calculate the distance between the origin and point A.



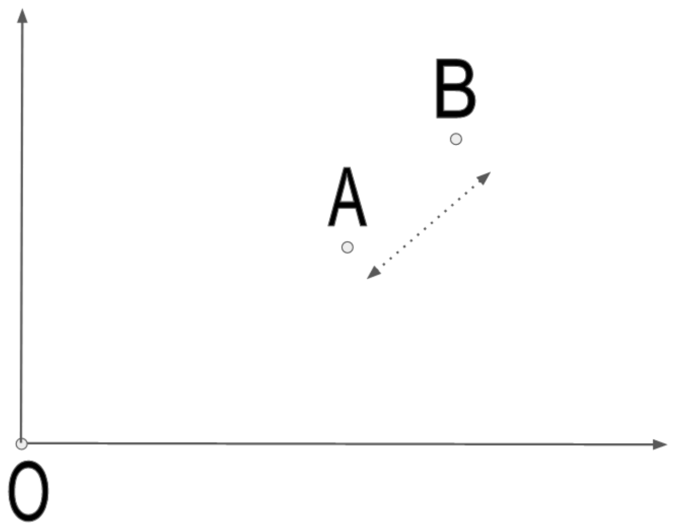
This comes out to be 3 units. In the same way, we can calculate the distance between the origin and point B



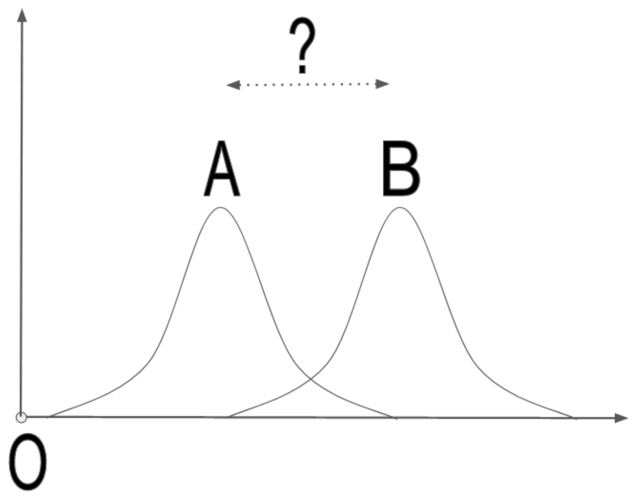
- which comes out to be 4 units. Now using these two quantities, we can calculate the distance between points A and B, which is 1 unit.



Simple enough? Lets extend this concept to a 2D plane



You can see that distance is still relevant as a way to compare two points. But can we use the same way of comparison on two probability distributions?



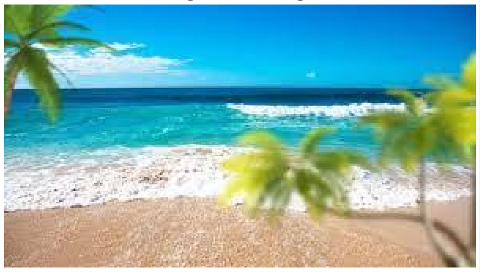
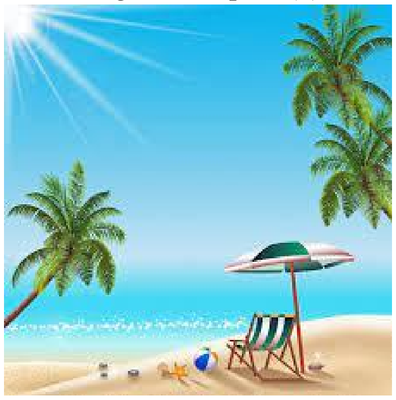
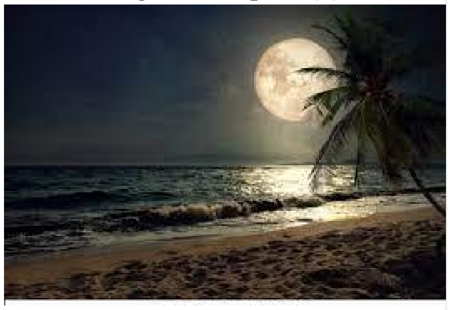
As you might have guessed, we need a more elaborate way to first describe, and then compare, the two probability distributions. This is where KL Divergence comes in, where **we can calculate how much information needs to be changed to obtain the first distribution from the second**. Mathematically, KL Divergence can be defined as [1]

$$\begin{aligned}\mathrm{KL}(p\|q) &= -\int p(\mathbf{x})\ln q(\mathbf{x})\,\mathrm{d}\mathbf{x} - \left(-\int p(\mathbf{x})\ln p(\mathbf{x})\,\mathrm{d}\mathbf{x}\right) \\ &= -\int p(\mathbf{x})\ln\left\{\frac{q(\mathbf{x})}{p(\mathbf{x})}\right\}\mathrm{d}\mathbf{x}.\end{aligned}$$

where, p(x) is usually the true or original distribution and q(x) is the model distribution which is getting compared to p(x). We will come back to the maths later, but first, let's understand KL Divergence with an example.

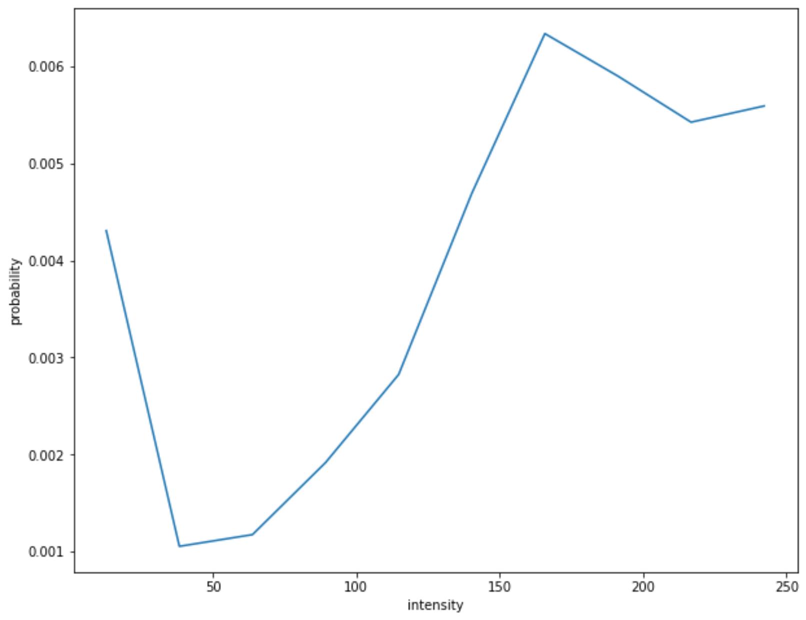
Example: Image Similarity

To understand the nitty gritty of the concept, let's take an example of image comparison. Given an image of a beach, and you want to compare which of them are visually similar.

Original image	Image to compare (1)	Image to compare (2)
		

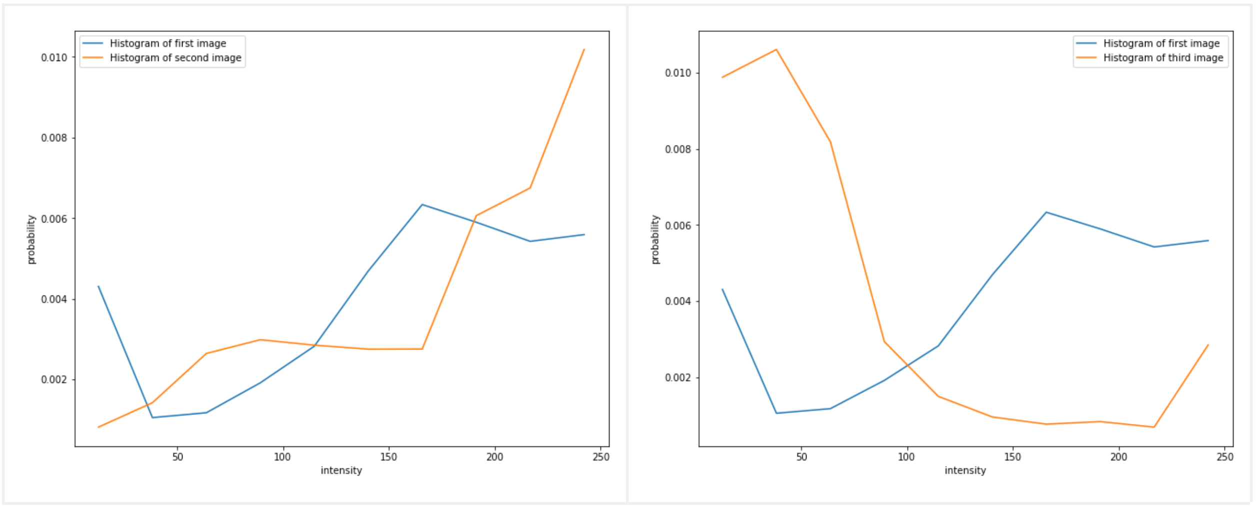
From the images, we can guess that the first two images are images of a beach taken during daytime, whereas the third one is taken during night time. One way to do a computational comparison is to calculate a histogram of pixel intensities of each image and compare them using KL divergence.

For simplicity, we do not consider the image size or the ordering of the content, but instead compare the pixel intensities. If we plot the histogram for our original image, it looks something like this.



From the graph, we can observe that the probability of this image having higher pixel intensities is higher, i.e. there are more whiter shades present in the image.

Now let us plot the histogram for our original image alongside the images we have to compare.



As expected, the histogram of the first image is more similar to the histogram of the second image, in comparison to the third. So just on the basis of these visualizations, we can conclude that the image_to_compare (1) is more similar to the original image. But can we prove this with a measurable quantity?

This is where KL Divergence can help. If we numerically calculate the KL Divergence between the first image and the second image, it comes out to be 0.008 [3]. Whereas the KL Divergence between the original image and the third image, it comes out to be 0.039. That is, lower the value of KL Divergence, the more similar two distributions are.

Mathematical Representation of KL Divergence

In this section, in order to demystify the complex concepts, I will try to get the overall understanding bit by bit and increase the complexity as we go.

1. Simple explanation of KL Divergence

As explained in the previous section, KL Divergence is a measure of comparison between two distributions wherein we calculate how much information needs to be changed to obtain the first distribution from the second. It was first introduced in the paper "On Information and Sufficiency" [10] by S. Kullback and R. A. Leibler in 1951.

2. Mathematical formulation of KL Divergence

The formula for KL Divergence is as follows [1]

$$\begin{aligned} \mathrm{KL}(p\|q) &= -\int p(\mathbf{x}) \ln q(\mathbf{x})\, \mathrm{d}\mathbf{x} - \left(-\int p(\mathbf{x}) \ln p(\mathbf{x})\, \mathrm{d}\mathbf{x}\right) \\ &= -\int p(\mathbf{x}) \ln \left\{\frac{q(\mathbf{x})}{p(\mathbf{x})}\right\} \mathrm{d}\mathbf{x}. \end{aligned}$$

where, p(x) is usually the true or original distribution and q(x) is the model distribution which is getting compared to p(x). Both p(x) and q(x) are considered to be continuous distributions

3. Relationship of KL Divergence with other metrics

Consider the formula -

$$\mathrm{KL}(p\|q) = -\int p(\mathbf{x}) \ln q(\mathbf{x})\, \mathrm{d}\mathbf{x} - \left(-\int p(\mathbf{x}) \ln p(\mathbf{x})\, \mathrm{d}\mathbf{x}\right)$$

If you are in ML space, you might easily recognize that the first term is cross entropy, which is frequently used as a loss function for training supervised deep learning algorithms. And the second term is just the entropy of p(x). Therefore

$$\begin{aligned} D_{\mathrm{KL}}(P \parallel Q) &= \sum_{x\in\mathcal{X}} p(x) \log \frac{1}{q(x)} - \sum_{x\in\mathcal{X}} p(x) \log \frac{1}{p(x)} \\ &= \mathrm{H}(P,Q) - \mathrm{H}(P) \end{aligned}$$

[2]

where, H(P, Q) is the cross entropy of P and Q and H(P) is the entropy of P

4. Extension of KL Divergence for discrete probability distributions

To apply KLD for discrete probability distributions, instead of integrating over a probability space, we consider the formula of KLD with respect to the discrete possible points of x

$$D_{\mathrm{KL}}(P \parallel Q) = -\sum_{x \in \mathcal{X}} P(x) \log\left(\frac{Q(x)}{P(x)}\right)$$

[2]

5. Unit for measurement of KL Divergence

If the logarithm has base *e*, KLD is measured in nats. Otherwise if the logarithm has base 2, KLD is measured in bits.

6. Derivation of KLD for two univariate gaussian distributions

Let's start with the original formula for KL Divergence and derive it for two univariate gaussian distributions

$$KL(P||Q) = -\int p(x)\log(q(x))dx - (-\int p(x)\log(p(x))dx)$$
$$= \int p(x)\log(p(x))dx - \int p(x)\log(q(x))dx$$

where p(x) and q(x) are the probability distribution functions of univariate gaussian distribution with means and variances as (μ_1, σ_1) and (μ_2, σ_2) respectively

From the definition of gaussian distribution, we know that pdf of a gaussian is

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}}e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

Let's calculate $log(p(x))$

$$p(x) = \frac{1}{\sigma_1\sqrt{2\pi}}e^{\frac{-(x-\mu_1)^2}{2\sigma_1^2}}$$

$$log(p(x)) = log(\frac{1}{\sigma_1}) + log(\frac{1}{\sqrt{2\pi}}) + (-\frac{(x-\mu_1)^2}{2\sigma_1^2})$$

$$log(p(x)) = -log(\sigma_1) + log(\frac{1}{\sqrt{2\pi}}) - \frac{(x-\mu_1)^2}{2\sigma_1^2}$$

Similarly for $log(q(x))$

$$q(x) = \frac{1}{\sigma_2\sqrt{2\pi}}e^{\frac{-(x-\mu_2)^2}{2\sigma_2^2}}$$

$$log(q(x)) = -log(\sigma_2) + log(\frac{1}{\sqrt{2\pi}}) - \frac{(x-\mu_2)^2}{2\sigma_2^2}$$

Replacing these values in formula,

$$KL(P||Q) = \int p(x)\log(p(x))dx - \int p(x)\log(q(x))dx$$

$$= \int p(x) * (-log(\sigma_1) + log(\frac{1}{\sqrt{2\pi}}) - \frac{(x-\mu_1)^2}{2\sigma_1^2}))dx - \int p(x) * (-log(\sigma_2) + log(\frac{1}{\sqrt{2\pi}}) - \frac{(x-\mu_2)^2}{2\sigma_2^2})dx$$

Because we integrating the equation over x, we can use these formulas which can be derived from the definition of gaussian distribution

1. $\int N(x|\mu, \sigma^2)dx = 1$
2. $\int x * N(x|\mu, \sigma^2)dx = \mu$
3. $\int x^2 * N(x|\mu, \sigma^2)dx = \mu^2 + \sigma^2$

Lets focus on the first term

$$\int p(x) * (-log(\sigma_1) + log(\frac{1}{\sqrt{2\pi}}) - \frac{(x-\mu_1)^2}{2\sigma_1^2})dx$$

$$= \int p(x) * (-log(\sigma_1) + log(\frac{1}{\sqrt{2\pi}}) - \frac{(x^2+\mu_1^2-2x\mu_1)}{2\sigma_1^2})dx$$

$$= (-log(\sigma_1) + log(\frac{1}{\sqrt{2\pi}}) - \frac{(\sigma_1^2+\mu_1^2+\mu_1^2-2\mu_1^2)}{2\sigma_1^2})$$

$$= (-log(\sigma_1) + log(\frac{1}{\sqrt{2\pi}}) - \frac{1}{2}))$$

Now let's focus on the second term

$$\int p(x) * (-log(\sigma_2) + log(\frac{1}{\sqrt{2\pi}}) - \frac{(x-\mu_2)^2}{2\sigma_2^2})dx$$

$$= \int p(x) * (-log(\sigma_2) + log(\frac{1}{\sqrt{2\pi}}) - \frac{(x^2 + \mu_2^2 - 2x\mu_2)}{2\sigma_2^2})dx$$

$$= -log(\sigma_2) + log(\frac{1}{\sqrt{2\pi}}) - \frac{(\mu_1^2 + \sigma_1^2 + \mu_2^2 - 2\mu_1\mu_2)}{2\sigma_2^2})$$

$$= -log(\sigma_2) + log(\frac{1}{\sqrt{2\pi}}) - \frac{((\mu_1 - \mu_2)^2 + \sigma_1^2)}{2\sigma_2^2})$$

Subtracting second term from the first

$$KL(P||Q) = ((-log(\sigma_1) + log(\frac{1}{\sqrt{2\pi}}) - \frac{1}{2}))) - (-log(\sigma_2) + log(\frac{1}{\sqrt{2\pi}}) - \frac{((\mu_1 - \mu_2)^2 + \sigma_1^2)}{2\sigma_2^2}))$$

$$= -log(\sigma_1) - \frac{1}{2} + log(\sigma_2) + \frac{((\mu_1 - \mu_2)^2 + \sigma_1^2)}{2\sigma_2^2}$$

Verification of this derivation using sympy can be found here [4]

7. Asymmetric nature of KLD

Let's try to understand this with the formula. We saw that

$$D_{KL}(P \,||\, Q) = H(P, Q) - H(P)$$

By that definition,

$$D_{KL}(Q \,||\, P) = H(Q, P) - H(Q)$$

We can see that the second term, H(P) and H(Q) should give us completely different values

8. Pseudocode for calculating KLD

Here you go! [3]

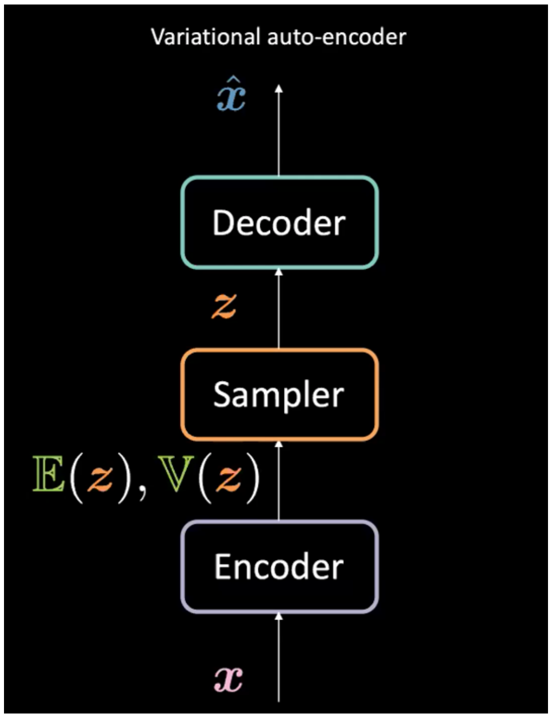
```
def kld(y_true, y_pred):
      """Computes KL divergence between `y_true` and `y_pred`.
      `loss = y_true * log(y_true / y_pred)`
      """
      return np.sum(y_true * np.log(y_true / y_pred))
```

Project Implementation

KL Divergence is the one of the most popular and effective metrics to quantify the similarity of two probability distributions. It has its use cases in a variety of fields such as computer vision, information systems, document retrieval and speech recognition to name a few [8]. It has also been used as a learning mechanism, for example, as a loss function to variational autoencoders (VAE). In this section, I will explore this specific application, wherein I will train a simple VAE with the help of KL divergence for an image generation task.

Let's formally define our problem statement. We train a deep learning model, more specifically, a multilayer perceptron based variational autoencoder to generate a 28x28 image resembling that of Fashion MNIST [9]. The brief explanation of the inner workings of VAE [11] is as follows -

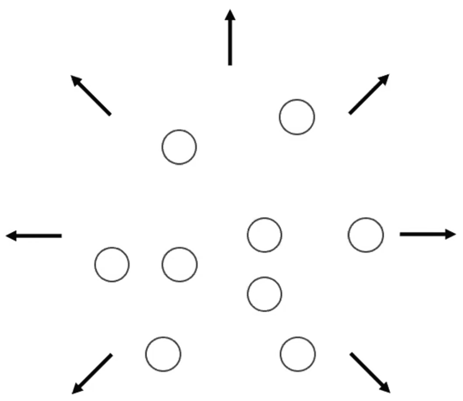
- VAE is made up of two parts, an encoder part and a decoder part
- The encoder part is responsible to approximate the mean and the variance of the training data
- Whereas the decoder part tries to reconstruct the image back from the output of our encoder
- To introduce variability in generation, we take a random sample from the mean and variance of the encoder before passing it to the decoder
- The loss function of VAE is comprised of reconstruction term (based on cross entropy loss) and regularization term (based on KL Divergence)



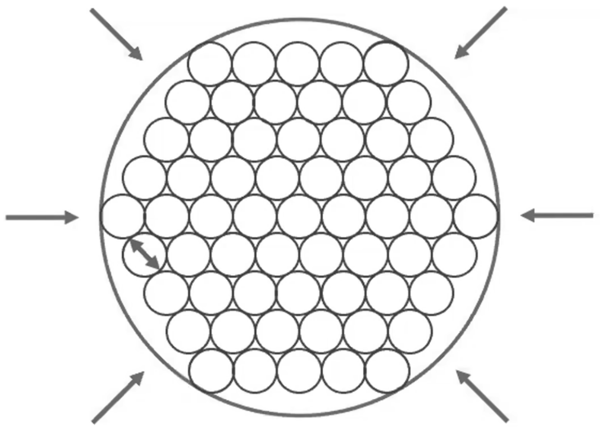
Let us expand on the use of KL divergence in VAE -

The default behaviour of the regularization term (i.e. without using KLD) is that it tends to define each image as a distinct entity, and in that pursuit, it fails to understand the relationship between the images in higher dimension.

Suppose the image below represents estimated values as circles, wherein the centre is the mean, and the radius is the variance [7]. The regularization term pushes the estimated region away from each other so much that the training collapses.

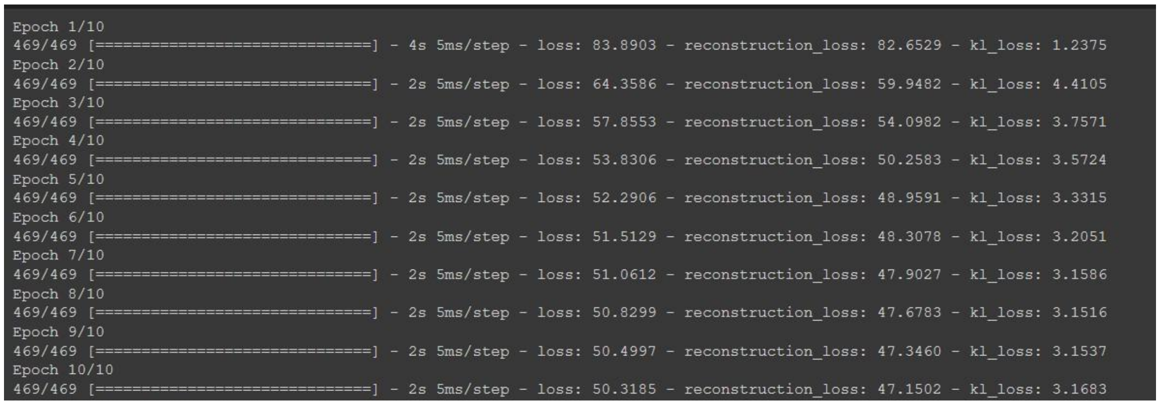


In order to penalize this, we introduce a regularization term which keeps the bounds intact. This is done by minimizing the KL Divergence between the estimated region and a univariate gaussian with mean 0 and variance as 1. This prevents the explosion of the loss function of VAE.



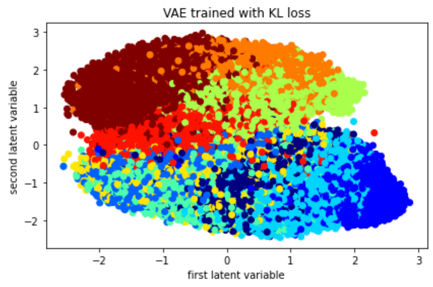
Results

From the description of VAE in the section above, I built a working model of VAE which uses KLD as a loss function. The architecture is based on a simple multilayer perceptron based VAE with two hidden layers with 100 neurons each and a latent dimension with 2 variables. It is then trained with Adam optimizer with learning rate as 0.0001 for 10 epochs. Here's how the training looks like

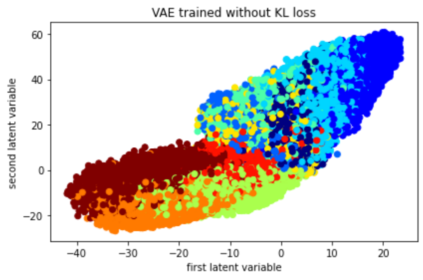


```
Epoch 1/10
469/469 [==============================] - 4s 5ms/step - loss: 83.8903 - reconstruction_loss: 82.6529 - kl_loss: 1.2375
Epoch 2/10
469/469 [==============================] - 2s 5ms/step - loss: 64.3586 - reconstruction_loss: 59.9482 - kl_loss: 4.4105
Epoch 3/10
469/469 [==============================] - 2s 5ms/step - loss: 57.8553 - reconstruction_loss: 54.0982 - kl_loss: 3.7571
Epoch 4/10
469/469 [==============================] - 2s 5ms/step - loss: 53.8306 - reconstruction_loss: 50.2583 - kl_loss: 3.5724
Epoch 5/10
469/469 [==============================] - 2s 5ms/step - loss: 52.2906 - reconstruction_loss: 48.9591 - kl_loss: 3.3315
Epoch 6/10
469/469 [==============================] - 2s 5ms/step - loss: 51.5129 - reconstruction_loss: 48.3078 - kl_loss: 3.2051
Epoch 7/10
469/469 [==============================] - 2s 5ms/step - loss: 51.0612 - reconstruction_loss: 47.9027 - kl_loss: 3.1586
Epoch 8/10
469/469 [==============================] - 2s 5ms/step - loss: 50.8299 - reconstruction_loss: 47.6783 - kl_loss: 3.1516
Epoch 9/10
469/469 [==============================] - 2s 5ms/step - loss: 50.4997 - reconstruction_loss: 47.3460 - kl_loss: 3.1537
Epoch 10/10
469/469 [==============================] - 2s 5ms/step - loss: 50.3185 - reconstruction_loss: 47.1502 - kl_loss: 3.1683
```

If we visualize the latent space, we can see that the model is able to separate the classes well even when it doesn't know the actual labels

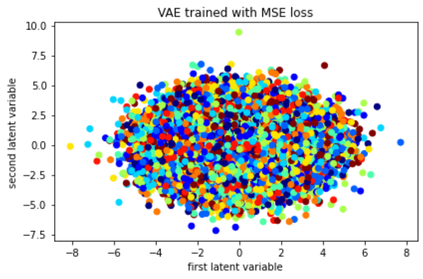


When we compare this with the latent space of a VAE trained without KL divergence loss,

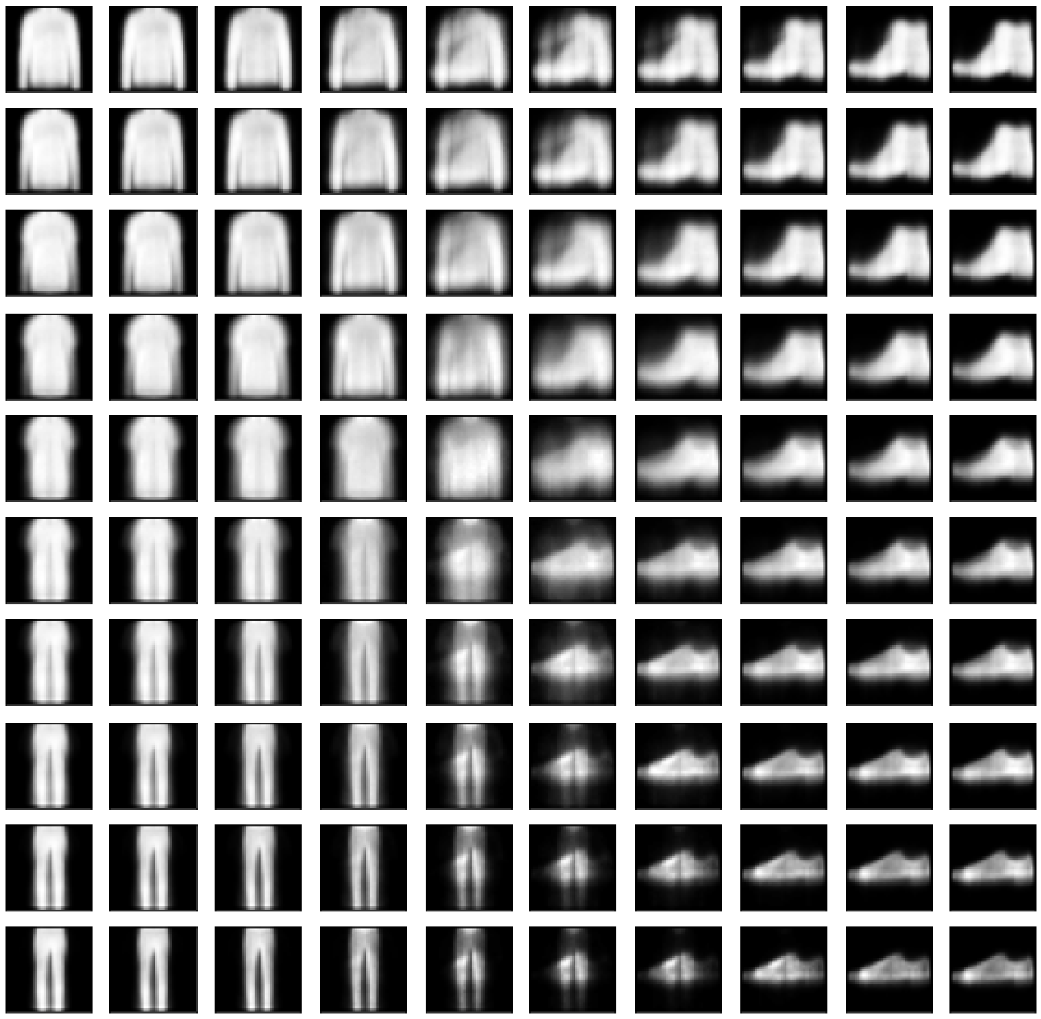


we see the same pattern as described in the previous section, i.e. if we train the model without KL loss, the latent space is highly distributed, ranging from -40 to 60, in comparison to a model trained with KL loss wherein the latent space lies in the range of -3 to 3.

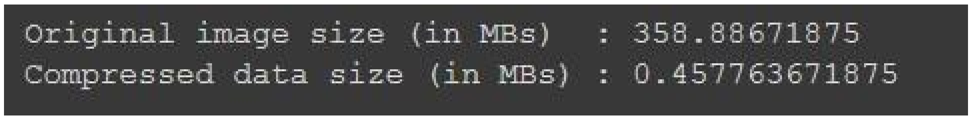
On the other hand, if we use euclidean distance as a training metric instead of KL loss, we see that it fails to differentiate between categories of data and enforces them to be the same (mean as zero and variance as 1), which beats our purpose of generating new images which belong in the same distribution as the data



Below are the generated images from the VAE model trained with KL loss



Finally, we compare the memory size of the original image and the compressed data, wherein we can see that there is significant decrease in memory size



```
Original image size (in MBs)  : 358.88671875
Compressed data size (in MBs) : 0.457763671875
```

The source code can be obtained from here [6]

Conclusion and Future Scope

This concludes the in-depth exploration of Kullback Leibler divergence along with its mathematical formulations, and derivation for two univariate gaussian distributions. I also laid out a practical use case of KL divergence as a loss function for Variational Autoencoders and proved its effectiveness. As discussed in the previous sections, KLD helps circumvent the gradient explosion during training and provides high quality data compression which is evident from the results. I believe that even when the fields of Artificial Intelligence and Data Science evolve, KL divergence will always be an important metric for learning because of its utility.

As a part of future enhancement, I would explore the implications of using a different architecture of VAE (perhaps which is convolutional based) to assess if KL divergence has a similar learning improvement. I would also explore the learning of KLD based VAE on a higher dimensional and more realistic data to accurately benchmark the performance.

Acknowledgement

I would like to thank Prof. Bukovsky for his guidance and suggestions throughout the project. I would also like to thank Prof. Jelínik for his valuable feedback for the VAE implementation. All the feedback I received helped me improve the quality of the project. I would also like to thank my colleagues who engaged in technical discussions for the project and who helped proofread the report.

References

[1] Bishop, Christopher M. "Pattern recognition." *Machine learning* 128.9 (2006).
[2] En.wikipedia.org. 2021. *Kullback–Leibler divergence - Wikipedia*. [online] Available at: <https://en.wikipedia.org/wiki/Kullback%E2%80%93Leibler_divergence> [Accessed 13 December 2021].
[3] GitHub. 2021. *expose_klDiv/KLDfromScratchNumpy.ipynb at main · faizankshaikh/expose_klDiv*. [online] Available at: <https://github.com/faizankshaikh/expose_klDiv/blob/main/code/KLDfromScratchNumpy.ipynb> [Accessed 13 December 2021].
[4] GitHub. 2021. *expose_klDiv/KLDinSympy.ipynb at main · faizankshaikh/expose_klDiv*. [online] Available at: <https://github.com/faizankshaikh/expose_klDiv/blob/main/code/KLDinSympy.ipynb> [Accessed 13 December 2021].
[5] GitHub. 2021. *expose_klDiv/POCofKLDiv.ipynb at main · faizankshaikh/expose_klDiv*. [online] Available at: <https://github.com/faizankshaikh/expose_klDiv/blob/main/code/POCofKLDiv.ipynb> [Accessed 13 December 2021].

[6] GitHub. 2021. *expose_klDiv/VAEfromScratch.ipynb at main · faizankshaikh/expose_klDiv*. [online] Available at: <https://github.com/faizankshaikh/expose_klDiv/blob/main/code/VAEfromScratch.ipynb> [Accessed 13 December 2021].
[7] LeCun, Y. and Canziani, A., 2021. [online] Youtube.com. Available at: <https://www.youtube.com/watch?v=7Rb4s9wNOmc> [Accessed 13 December 2021].
[8] Ji, Shuyi et al. "Kullback-Leibler Divergence Metric Learning." *IEEE transactions on cybernetics*, vol. PP 10.1109/TCYB.2020.3008248. 28 Jul. 2020, doi:10.1109/TCYB.2020.3008248
[9] Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. Han Xiao, Kashif Rasul, Roland Vollgraf. arXiv:1708.07747
[10] Kullback, S.; Leibler, R.A. (1951). "On information and sufficiency". Annals of Mathematical Statistics. 22 (1): 79–86. doi:10.1214/aoms/1177729694. JSTOR 2236703. MR 0039968.
[11] Kingma, Diederik P.; Welling, Max (2014-05-01). "Auto-Encoding Variational Bayes". arXiv:1312.6114