Analysis on isometry as an impact on the stability of neural network training

Kevin Shi and Shuxiao Chen

December 20, 2020

Abstract

Isometry in convolutional neural networks is a relatively new concept; however, many assorted other regularisation techniques, such as batch normalisation (BN), have existed in the deep neural network community for a relatively long period of time. In this work, we demonstrate that isometry and batch normalisation are in fact very similar and may be a part of a larger form of regularisation defined as normalisation.

1 Introduction

Over the previous decade, much headway has been made in image processing with neural networks, from multi-layer perceptrons (MLP) to VGG to today's ResNet architectures, computer vision has become more and more accurate. However, most of these gains are empirical, with only a surface level understanding of why specific techniques improve performance. This is especially evident in deep networks, where many techniques exist to prevent vanishing/ exploding gradient, such as batch norm and skip connections, but despite their popularity and wide-found practical success are still poorly understood.

This project originally started with the goal of being able to compress and Bayesian fuse semantic segmentation maps, which we define as $(H \times W \times N)$ matrices where H is height, W is width, and N is the number of classes. Thus, a Variational AutoEncoder (VAE) architecture was selected for our use case. Isometry applied to the encoding half of the network seemed extremely suited for our purposes. Isometry is a map of a space onto another such that the distance, and by consequence information, is preserved in the second space. This means that with isometry as a property, hopefully the encoder will be able to discover the low-dimensional model of our high-dimensional matrix. However, along the way, we found interesting properties of isometry in networks that imply isometry may have the same training properties as BN, namely loss and gradient smoothness.

This paper's goal is to attempt to explain/understand one of the most recent successes in an attempt to find an alternative to the currently popular ResNet architecture, isometry as implemented in IsoNet [6]. We attempt to explain

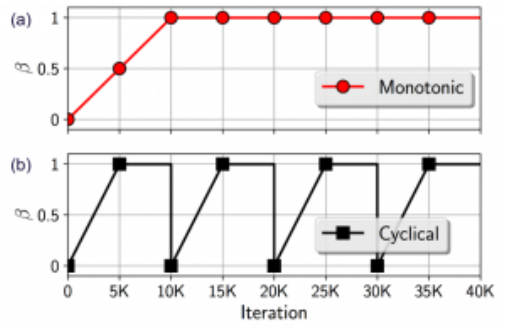


Figure 1: Monotonic and cyclic loss from [1]

isometry's success by demonstrating that the regularisation of isometry is similar to BN, and BN may in turn be implying isometry.

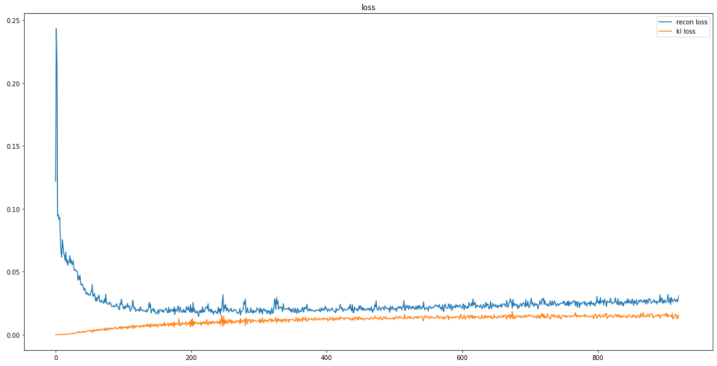
2 Related Works

Much of what we are investigating here are the properties of isometry on convolutional neural networks. Our specific implementation of isometry regulation spawns from *Deep Isometric Learning for Visual Recognition* by Qi, et al. [6]. Specifically, we utilise the three aspects of isometry that they implemented, delta initialisation, SReLU activation, and convolutional orthogonality constraint. Specifically, we apply these isometric properties to a VAE [3]. We attempt to explain why isometry is so effective in training by investigating the common problems and solutions in deep networks, such as vanishing/exploding gradient.

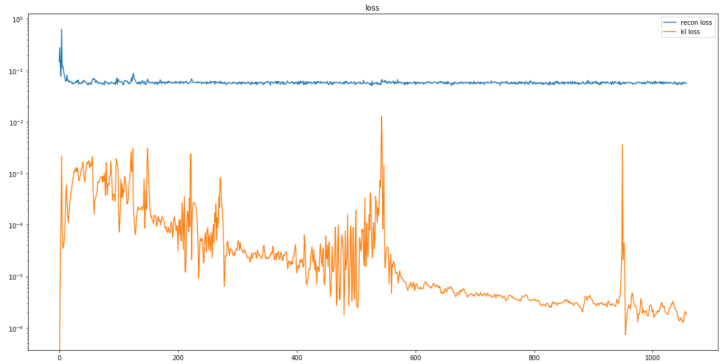
There are many articles attempting to investigate and explain exploding/ vanishing loss [2, 5], however, many focus on simple MLPs, and we find it easier to explain isometry's success through BN. The aforementioned articles also touch on BN, but the most helpful parallel we found in *How Does Batch Normalization Help Optimization?* by Santurkar, et al. [7].

This paper [7] seeks to understand why BN is effective, and they find that instead of reducing internal covariate shift as popular belief dictates, it instead describes the benefits of BN from providing "Lipshitzness"[1] of the loss function, smoothing the loss and gradients of the function. They also demonstrate that it is the normalisation property of BN that is responsible for this smoothing property, and any normalisation technique yielded similar results. These benefits are well-documented, and include gradient regularisation, ability to use higher learning rates (LRs), faster training, simplification of network design, and many others.

[1] L-Lipshitzness smooth: $|f(x_1) - f(x_2)| \leq L||x_1 - x_2||$, for all x_1 and x_2



(a) ResNet with SReLU, no BN



(b) ResNet with SReLU, no BN

Figure 2: Loss graphs for SReLU

3 Architecture

Our architecture consists of a VAE with a ResNet-18 architecture for encoding, and a structure similar to EDSR [4] with progressive upsampling for decoding. We also modified both encoder and decoder to be able to handle the isometry initialisation and constraints found in [6] by adding flags to enable/disable BN, change ReLU to SReLU, and change Kaiming/Delta initialisation; however, we will only utilise isometry for the encoder half of the network whilst leaving the decoder constant. We are also utilising cyclic loss introduced by Microsoft [1], which can be seen in Figure [1]. For the following experiments, we trained on the MNIST dataset.

Architecture	Reconstruction	KL
(a)	.0223	.0142
(b)	.0226	.0140
(c)	.0572	7.5×10^{-5}

Table 1: (a) ResNet; (b) ResNet with SReLU, no BN; (c) ResNet with no BN

(a) SReLU, no BN (b) No BN

Figure 3: Outputs of modified ResNet, top half is input, bottom half is output

4 Experiments and Results

4.1 SReLU

After running some experiments with SReLU, we noticed that the ResNet encoder network modified with SReLU and no BN was performing much better than ResNet with ReLU and no BN. As you can see in Table 1, we can see the modified SReLU with no BN performed practically the same to ResNet within acceptable margins, ResNet with no BN had a degenerate reconstruction KL loss divide. Why does this happen? If we think about this in a non-isometric sense, BN parametrises the underlying optimisation problem to smooth the underlying loss function [7]. Once our algorithm is caught in the local minimum of reduction of KL loss, it is extremely easy to start reducing the KL loss even more, making the reconstruction loss even more difficult to reduce, keeping the algorithm in the local minimum, as seen in Figure 2.

However, if we contextualise with isometry, the reason is due to the fact that reducing the KL divergence is inherently at odds with isometry - KL encourages outputs to be normal Gaussian, leading to all images to be the same; isometry wants to preserve projection distance, preventing all the projected distributions to be different. Even with only SReLU initialisation, we start to see this property. Conversely, this also goes to show that BN is enforcing isometry.

This realisation allows us to train at much higher KL weights using isometry constraints without the problem that we see in Figure 2.b. This may not be desirable for all use cases because we are now not emphasising reconstruction loss as much, but is extremely useful in other cases where we need high interpolation between classes, which is arguably the main goal of a VAE.

4.2 Isometry and Normalisation

Our hypothesis as to why isometry promotes network and gradient stability is because it is essentially providing the benefits of reparametrising the underlying optimisation problem to make its landscape significantly smoother as described in [7]. See Related Works for more information.

We claim that isometry is also in effect a normalisation. We can see in Figure 4 that isometry provides a much smoother loss landscape compared to just BN especially in the beginning of training, and also in the same 3000 time steps converges to a similar loss (seen in Table 2), mirroring the results found in [7] of the effects of normalisation. We can also see that the purely vanilla network without BN or isometry has such an unstable start that it falls into an undesired local minimum, only decreasing the KL loss and therefore outputting poor reconstruction loss, which is undesirable in any form of VAE. From this demonstration, we can show that isometry promotes stability and predictiveness in the loss and therefore the gradients - showing that isometry, like BN, is a form of normalisation in the network and yields normalisation's positive effects. The more important idea is that we have demonstrated that isometry provides a perhaps even stronger normalisation effect than just BN in specific circumstances.

However, the downside is that we have yet another hyperparameter to tune. We found that if we set isometry's orthogonality weight too low, we had the same problems as if we had a vanilla network, and if the weight was too high we would not train properly. Even so, this may be a positive for some networks - as with isometry regularisation/normalisation, unlike BN, we are able to tune the amount of isometry with one hyperparameter, allowing us more control over the network in more specific use cases.

Architecture	Total	Reconstruction	KL
Vanilla	.0550	.0550	5.8×10^{-7}
BN	.0469	.0340	.0129
Isometry	.0473	.0352	.1210

Table 2: Final losses of Figure 4

4.2.1 Justification

According to the definition,

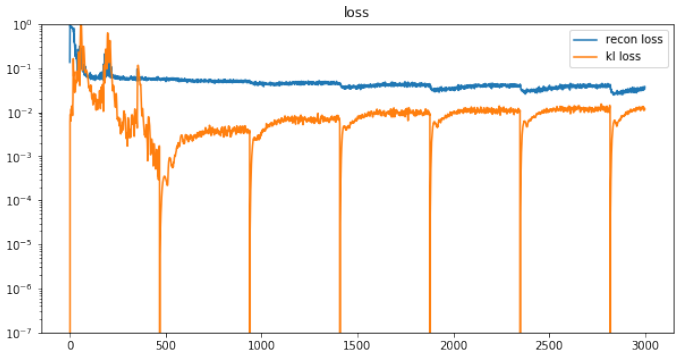
$$d_Y(f(a), f(b)) = d_X(a, b)$$

an isometry is a transformation which maps elements to the same or another metric space such that the distance between the elements in the new metric space is equal to the distance between the elements in the original metric space.

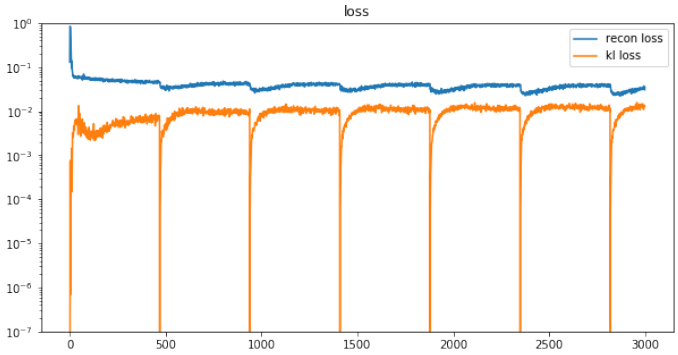
For instance, in an inner product space, the above definition reduces to

$$\langle v, v \rangle = \langle Av, Av \rangle$$

(a) Vanilla network (no BN, no isometry): Fails to converge, caught in local minimum of almost 0 KL loss



(b) Vanilla network with BN: Unstable start but convergence



(c) Vanilla network with isometry constraints: Stable start and convergence

Figure 4: Loss landscapes with modified vanilla encoder networks Encoders all have no skip connection. z/latent size is 20. Decoder is held constant with no BN, with skip connections. Loss/Optimiser (ADAM) is same across all plots with a LR of 0.0015, a KL weight of 0.004 with cyclic β.

for all $v \in V$, which is equivalent to saying that $A^*A = \mathrm{I}_V$. This also implies that isometries preserve inner products, as

$$\langle Au, Av \rangle = \langle u, A^*Av \rangle = \langle u, v \rangle .$$

Linear isometries are not always unitary operators, although if orthogonality is forced, the magnitude of the determinant of any orthogonal matrix is 1. This follows from basic facts about determinants, as follows,

$$1 = \det(I) = \det\left(Q^*Q\right) = \det\left(Q^*\right)\det(Q) = \big(\det(Q)\big)^2 .$$

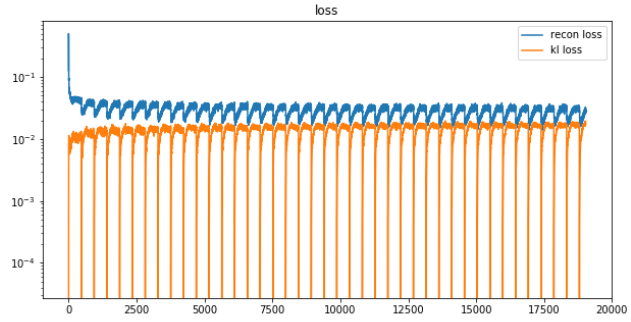
Geometrically, determinant can be viewed as the area scaling factor of the linear transformation described by the two-dimensional matrix, thus scale-invariant operations.

On the other hand, e.g. the standard score type of normalisation offers a space manipulation, $\frac{X-\mu}{\sigma}$, which can be decomposed to translational shift and scale. As isometry keeps the norm between metric spaces unchanged, this trait is analogous to the translational shift stage of normalisation. Also, as proved the scale-invariant property of matrices with a determinant whose magnitude is 1, isometry brought by forcing orthogonality essentially forces this second scale stage in normalisation.

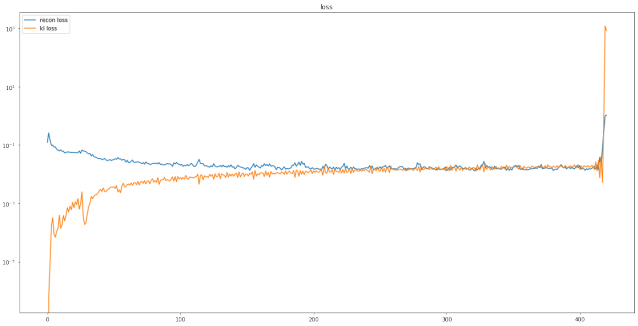
In summary, we found that whilst the reason for applying batch normalisation to the input to the network being effective in improving the training process by rearranging the input to have a roughly consistent mean and scale, the equivalent representation expressed in the weights of the network of such an operation is indeed the enforcement of orthogonality, or isometry during the initialisation, training and activation stages.

4.3 Catastrophic Loss

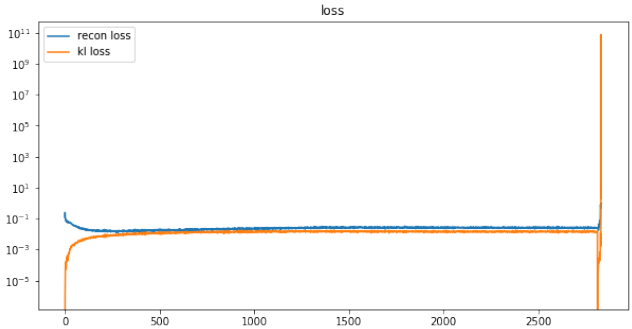
While training this network, we also see catastrophic explosion in loss leading to catastrophic failure. This failure is due to a sudden change in the slope of loss. We are using cyclic loss [1] to help prevent KL from vanishing, so changes in loss are common. Cyclic loss introduces at value β that slowly increases from 0 to 1, saturates, and then drops immediately from 1 to 0 as a soft reset (see Figure 1). KL loss weight is directly multiplied by β. After experimentation, we do not see any correlation between KL weight and loss, but we do see that increasing isometry weight directly correlates and increasing LR indirectly correlates to more stability. We think that the reason is due to exploding gradients, and high LR being one of the causes and the footprint of the loss graphs. These catastrophic failures tend to happen when we are cycling β from 1 to 0, but also happen randomly or at monotonic loss changes (see Figure 5). If this is true, it means that isometry helps promote network and gradient stability.



(a) Normal loss with cyclic β



(b) Failure at loss transition from increasing to constant β at iteration 470



(c) Failure at loss transition from 1 to 0 β cyclic reset at iteration 470

Figure 5: Loss graphs and catastrophic loss explosion with ResNet architectures

5 Conclusion

Although we attribute the exploding loss to exploding gradients, the catastrophic losses graphs here look quite suspicious. It may just be the case that we failed to implement what we wanted properly. After a thorough search through the architecture and many unit tests, we could not find anything that seemed like it could lead to the catastrophically exploding loss that we had. However, even if we disregard our results with exploding losses, we think that our other (first) experiments with BN and isometry show that our hypothesis of isometry provides normalisation and its benefits have merit.

5.1 Difficulties

Irregardless though, we did not reach our original objective in the allotted time, meaning that our planning was sub-optimal. One reason we did not accomplish what we sought out to do was that we spent too much time on non-essential functionality. For example, we spent a lot of time ensuring that the network architecture was flexible. We developed a network that could take arbitrary square image sizes and any amount of channels. We also added parameters to be able to change the number of ResNet stages and the number of layers within each stage, testing to ensure functionality. This was in preparation for moving from our MNIST training set to our semantic map training set. Although it is good programming practice to ensure the flexibility of the code, when there is a deadline sometimes it is better to be agile and firstly produce a minimum viable product instead of worrying about how easy to modify the code will be in the future, especially if one realises that time is short. We even finished setting up our semantic network and gathered some training data. We think that the communication between two people who originally did not know each other and the distanced COVID semester also promoted to this trouble. However, other classmates also must have had this problem, so it is not a valid excuse. After we found some interesting phenomena (our loss graphs), we decided to pivot to trying to explain and analyse the observations instead.

References

[1] Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Celikyilmaz, and Lawrence Carin. Cyclical annealing schedule: A simple approach to mitigating kl vanishing, 2019.

[2] Boris Hanin. Which neural net architectures give rise to exploding and vanishing gradients?, 2018.

[3] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2014.

[4] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2017.

[5] George Philipp, Dawn Song, and Jaime G. Carbonell. The exploding gradient problem demystified - definition, prevalence, impact, origin, tradeoffs, and solutions, 2018.

[6] Haozhi Qi, Chong You, Xiaolong Wang, Yi Ma, and Jitendra Malik. Deep isometric learning for visual recognition. In *ICML*, 2020.

[7] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization?, 2019.