Deep Neural Networks and Information Bottleneck

Mohit Gsoyal 2015EE10111 IIT Delhi
ELV832
Assignment
Sumeet Agarwal

November 17, 2018

Abstract

Obective: To understand and analyze the work done by Ravid Schwartz-Ziv and Naftali Tishby in "Opening the blackbox of Deep Neural Networks by information", and try to validate and extend on their experimental observations.

1 Introduction

Tishby proposed the Information Bottleneck Tradeoff [1] providing a method to optimally compress the input X and predict Y. In mathematical terms, if we $t \in T$ are the compressed representations of input X which help in prediction of Y, the labels. Then the following markov chain is formed $Y \rightarrow X \rightarrow T$. Now, to achieve the optimal compression in X while preserving maximum information about the labels Y is written as the following optimization problem,

$$\min_{p(t/x),p(y/t),p(t)} \{I(X;T) - \beta T(Y;T)\} \tag{1}$$

where, the probabilities $p(t/x), p(y/t), p(t)$ are to be optimized, and $I(.,.)$ represents the mutual information, and β denotes the penalty term controlling the tradeoff betwen compression and prediction accuracy.

Tishby in his latest work [2] experimented on evolution of the mutual inforation of layers with the input and the ground truth for a standard neural network. His work revolves around these curves, which he mentions as *"DNN information paths in the Information Plane"*.

2 Data Generation

All the experiments done in this report are performed on the same dataset as described in [2]. The task in of binary classification which are invariant under $O(3)$ rotaions with 12 inputs representing 12 uniformly distributed points on a $2D$ sphere. As a result 4096 input patterns are genrated. For generating labels one can look at the Experimental setup in [2].

3 Experiments and Discussion

3.1 Ambiguity: Dataset for MI calculation

In [2], while calculating the mutual information (MI) curves for the above dataset, there is no explanation or comment on which data was uses to plot the curves. They do mention the dataset used for training the DNN, but which datasection was used for MI calculation has been skipped from their script. This is important in two ways,

1. To measure generalization, if only the training dataset is used, it is impossible to ever encounter overfitting. One, however, may infer some form of underfitting though.

2. If the percentage of training data in the data used for plots is high, the curves would end up showing an obvious higher MI value with the output label Y.

Therefore, we plot separate curves for training and testing dataset. We experiment with two activation functions, ReLU and tanh with deafult binsize of 0.07 (calculated from 30 bins between -1 to 1) and the model size used is 10-5-3. The models below are trained for 4000 epochs and their Information paths for these epochs are plotted.

3.1.1 80 % Training Data and 20 % Testing Data

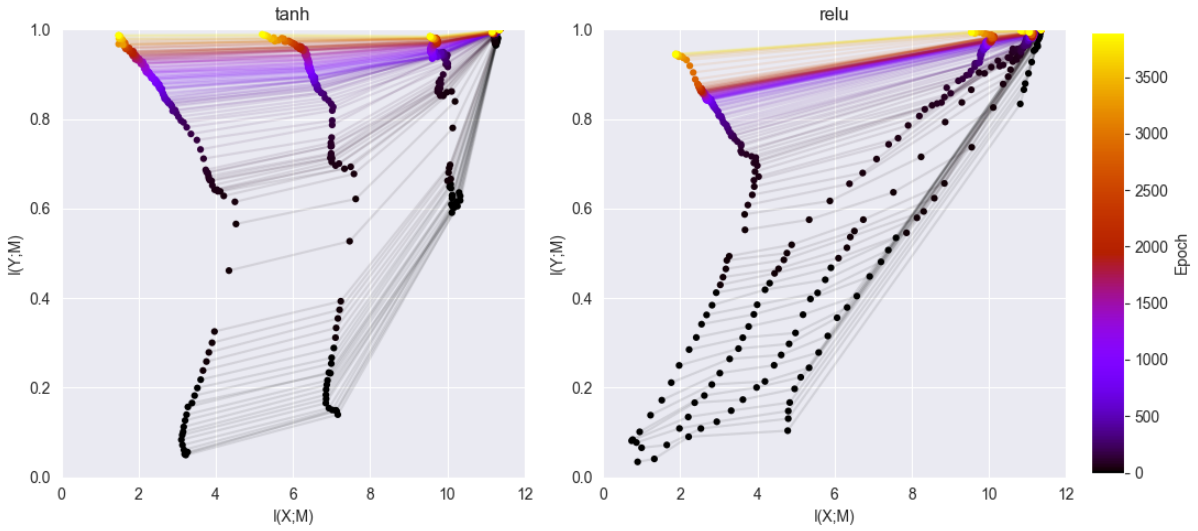

Figure 1: : The above curve is for training dataset only.

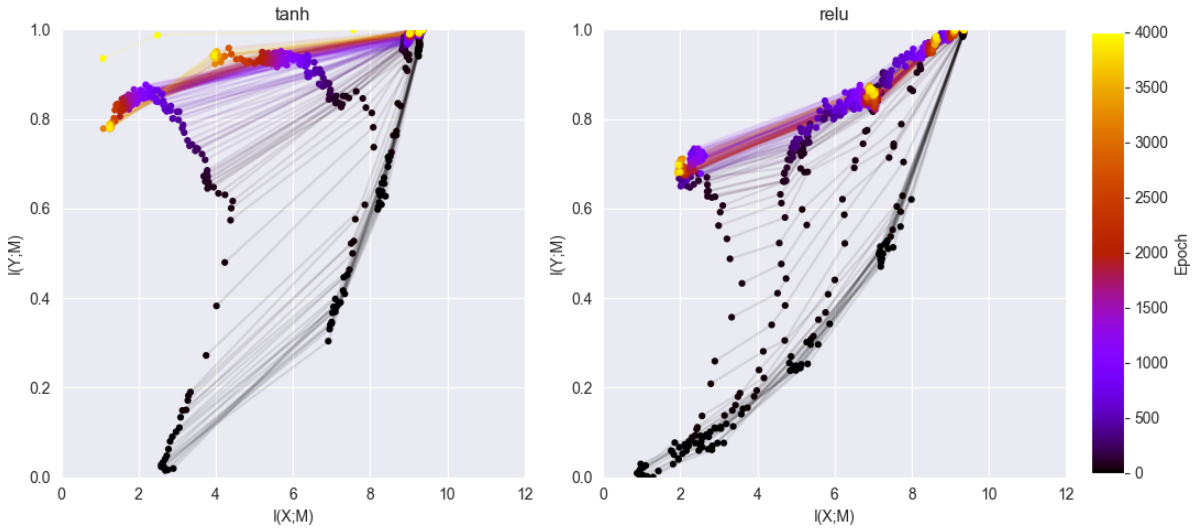

Figure 2: : The above curve is for testing dataset only.

The training accuracy of the models above are around 99% and about 85-90% testing accuracies are observed. One observation here is that, mutual information with input $I(T;X)$ is indeed an important measure because it is clear from the Figure 1 and Figure 5 that even for the penultimate and previous layer, the $I(T;X)$ are very similar for training and testing curves. However, for relu activation there is some disparity between the two, although not much at the final epochs.

3.1.2 20 % Training Data and 80 % Testing Data

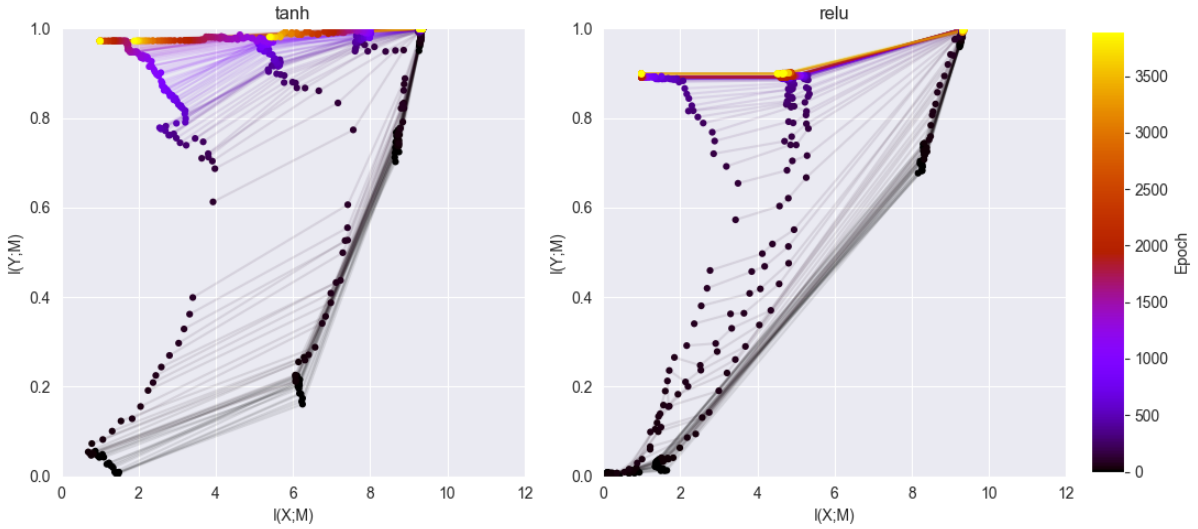

Figure 3: : The above curve is for training dataset only.

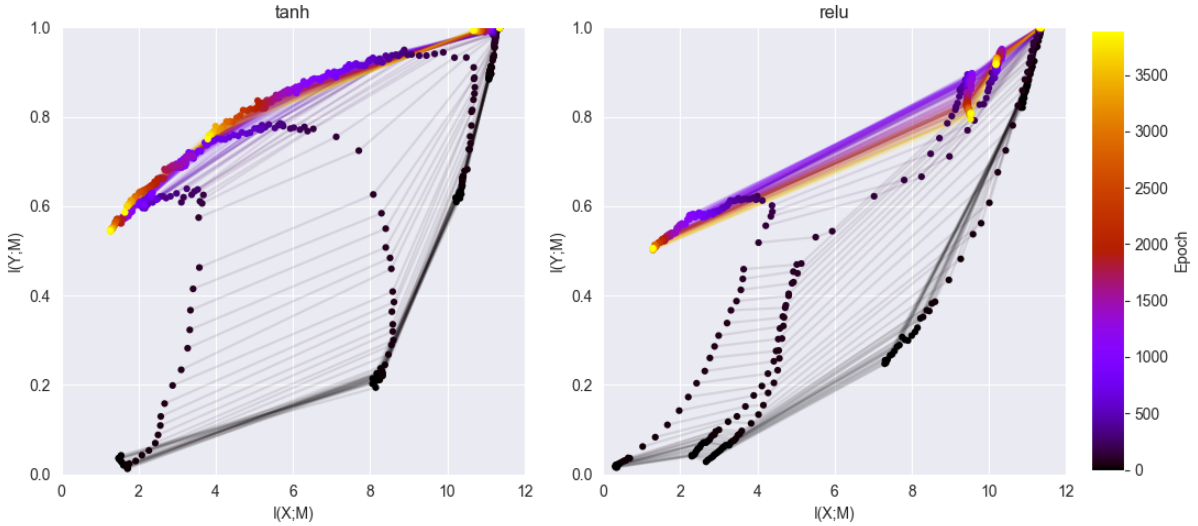

Figure 4: : The above curve is for testing dataset only.

The training accuracies for the above models were 99% but testing accuracies were almost 35-40% for both the activations. This is again affirmed from the figures the importance of $I(T;X)$ for the amount of generalization acheived. For the tanh activaion, in the case of testing, there is a sudden increase in compression after around 1000 epochs in contrast to the training curve. On the other hand, for relu, exactly opposite is seen for the prefinal layers.

3.1.3 Ambiguity in calculation of Mutual Information

One more inference can be drawn from the above figures in accordance to the argument given in [3], there is no compression phase encountered in the case of relu. This phase is mentioned multiple times in [2], and was shown essential for generalization. We encounter similar values of generalization for both the cases but the amount of compression for prefinal layers is not even comparable. In the openreview, heated discussion follows Tishby uploading the MI curves for ReLU highlighting the problematic MI calculation done by [3]. This is the uploaded curve,

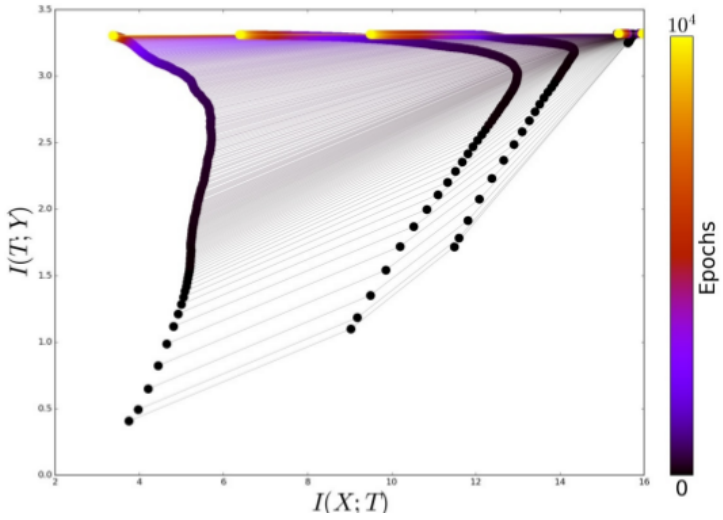

Figure 5: : Inforation Trajectory for Cifar10 CNN using ReLU.

This apparent difference in the curve is mainly due to the method of binning, because tishby is using binning only in his official code before calculation the probability distibutions which are then use to calculate MI. We look at four methods of binning listed below. Note that these experiments are only performed for tanh activation for 4000 epochs:

1. **Constant Bin size:** In this case, the size of the bin size is varied from 0.07 to 0.30 and the correspoding Trajectories are plotted in the Figure 6, 7, 8. We observe that with the increase in bin size more and more number of points get clustered into the same bin. This results in an apparent compression resulting in a shift towards the left for each epoch. We also observe that this size in inappropriate for different epochs because the graphs appears to become almost like a vertical line. This is only due to incorrect binning. Hence a more sophisticated binning must be used.

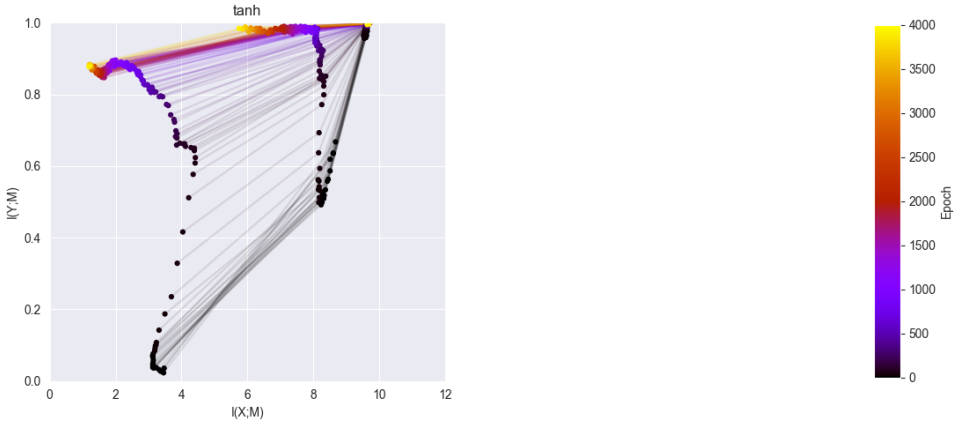

Figure 6: Constant Bin Size (c) 0.14.

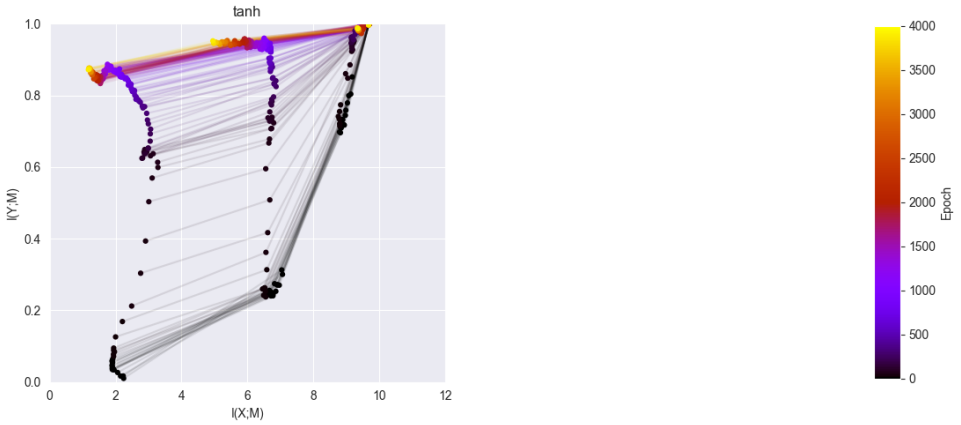

Figure 7: Constant Bin Size (c) 0.14.

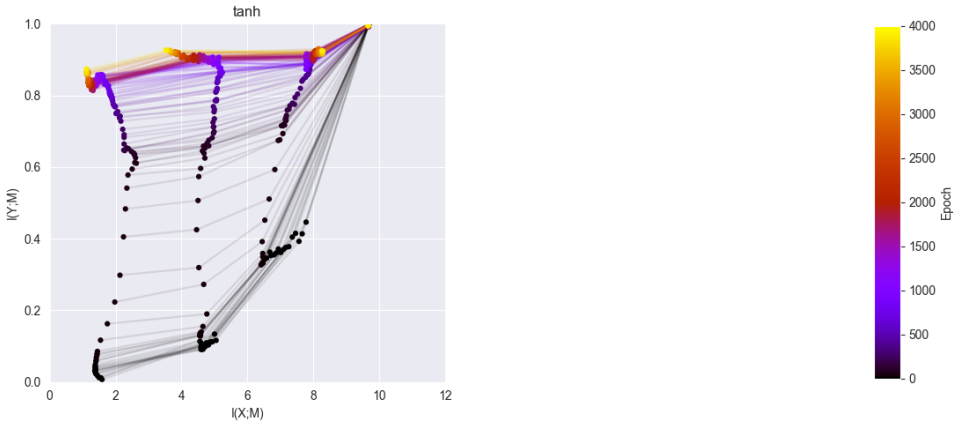

Figure 8

Figure 9: Constant Bin Size (c) 0.30.

2. **Min-Max at each layer:** In this case, we calculate the min-max at each epoch for every layer. This is then divided into k number of bins, this is more appropriate because if we consider scaling of input, then a constant bin size would badly fail to give reasonably accurate mutual information. Since $I(X, r.X) = H(X)\ \forall\ r \in \mathbb{R}\ and\ r \neq 0$. This equation will hold more perfectly in the above method because min max will appropriately scale the bin size (Figure 10, 11).

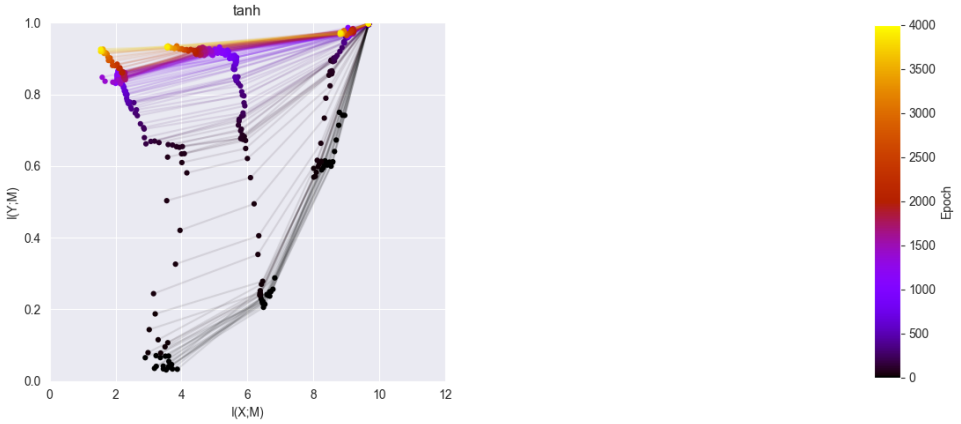

Figure 10: (a) No. of bins = 10

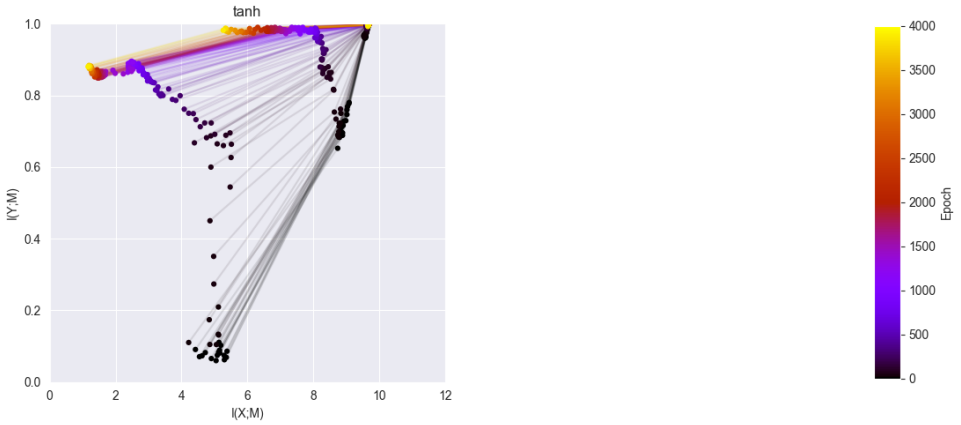

Figure 11: (b) No. of bins = 30

3. **Standard Deviation:** Standard deviation is also a good measure and seeming most appropriate to calculate the bin size. If the distribution is gaussian, then min-max may give wrong results depending on the sampling. But standard deviation will always be similar and hence a more robust method to evaluate bin size. Binsize is now defined as $binsize = multiplier * std(layer\ activations)$. Three multipliers are tested with as shown in the figures 12,13,14 below

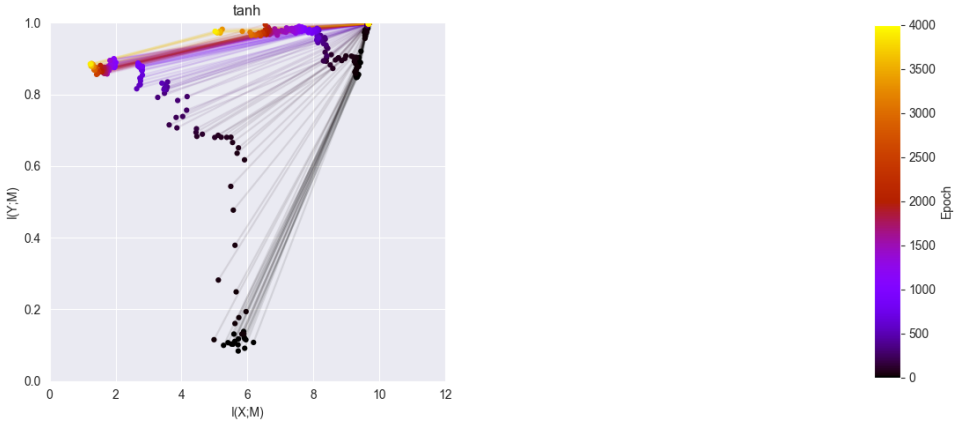

Figure 12: (a) Multiplier = 0.1

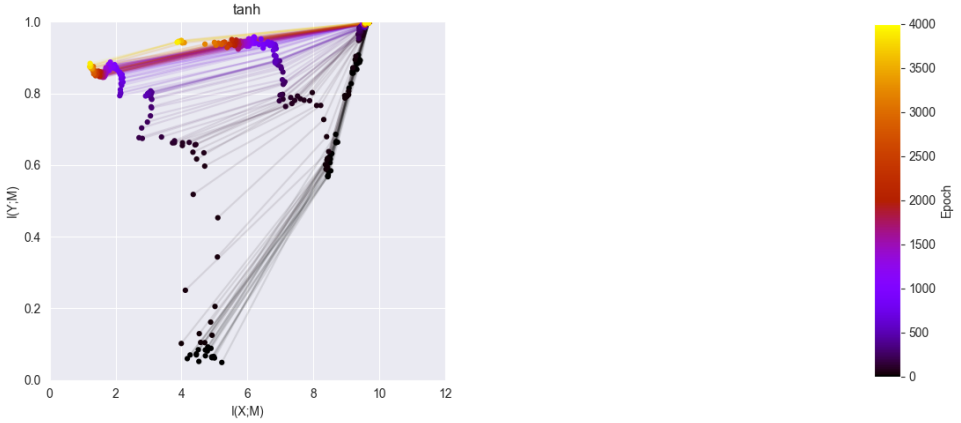

Figure 13: (b) Multiplier = 0.2

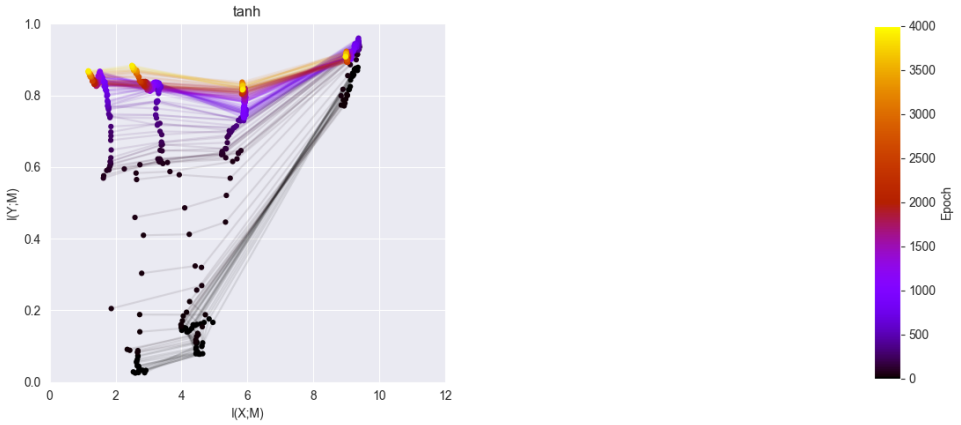

Figure 14: (b) Multiplier = 1

4. **Kernel Density Estimation (KDE):** On the lines of [3], we calculate the KDE upper bound which is the current best method for calculating MI available to us. We see that standard deviation gives a curve very near to the KDE upper bound which was expected given the robustness of standard deviation as a measure. Below is the curve for KDE upper bound 15,

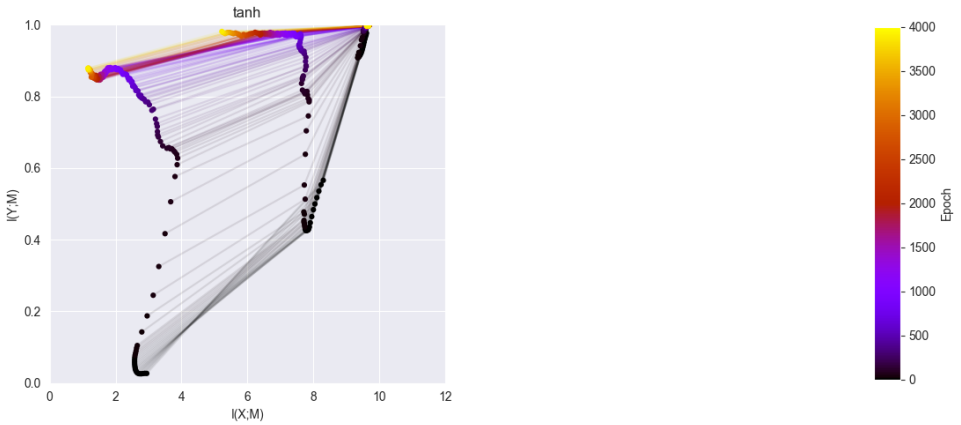

Figure 15: Kernel Density Estimation Upper Bound

Finally we conclude from these experiments that standard deviation is a better estimate for calculating MI, which should be done at every epoch and for each layer. A global value would give inaccurate values for mutual information and hence affect the validity of our deductions from the trajectories. Therefore there is still need to check if these methods can be well trusted before carrying out the experiments and drawing inferences from them. All the compression phases are lost when bigger in sizes are used, which somewhat matches the saturation arguments for tanh activation given in [3].

3.2 Effect of Batch Normalization:

Batch normalization [4] the most effective method for improving generalization working on most of the tasks deserves to be analyzed here. Below is the change in curve when batchnormaliztion is applied. We only plot the activation layers and not the intermediate output before the activation.

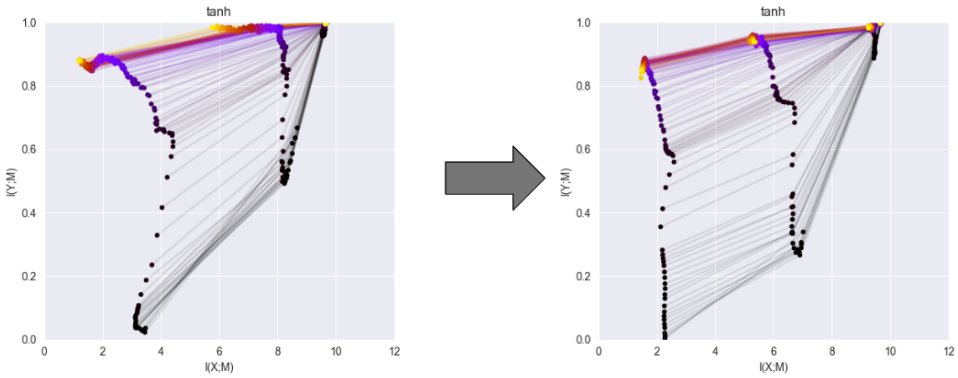

Figure 16: Batch Normalization effect with bin size = 0.07

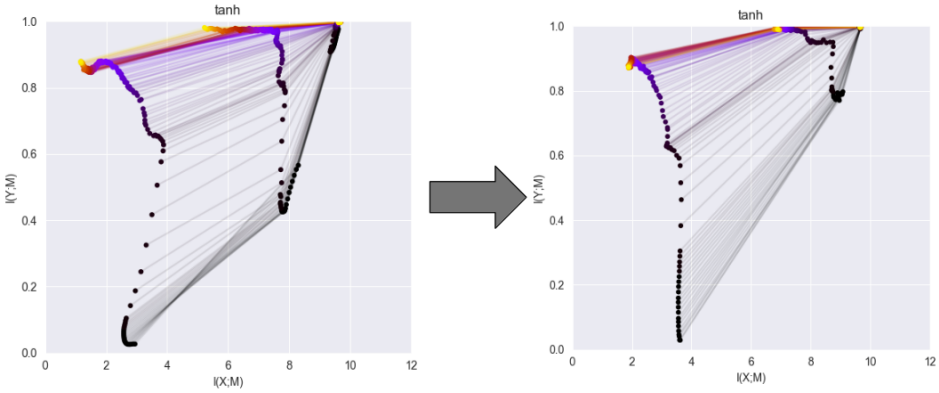

Figure 17: Batch Normalization effect with KDE upper bound

Since, batchnormalization changes the mean and variance of each layer and every neuron, it is obvious that we see a shift in the trajectory because the bin size is kept same. But the variance is changed. Here, we observe a compression from the very start of the training. This might point to acceleration which is the whole purpose of batch norm. We also observe that the distinction between the error minimization phase and compression phase kinds of fade for all the layer. This effect is most dominantly seen in the final layer. I would again point at reconsidering the validity of the MI calculation methods. It may turn out in future, that all these deductions were only due to the way of calculating MI and not solely due to architecture changes or other changes done in the experiments in this report.

4 Conclusion

With various experiments and comparing theoretical expectations with experimental observations, we pointed out ambiguities in the central paper being discussed here. We extensively experiment on the binning methods and compare the results with the state of the art, Kernel Density Estimation method for MI estimation. Though we fail to reproduce the result given by tishby (`https://www.dropbox.com/s/6aotykw6py37z1h/Naftali%20Tishby%20and%20Ravid%20Shwartz%20Ziv-final%20comment.pdf?dl=0`). It would still be interesting to find out if there is an actual compression phase which gives or points to compression. Our experiments with batch normalization also highlights the importance of compression in the trajectories which comes into appearance from the very start of training.

References

[1] Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method. *In Proceedings of the 37-th Annual Allerton Conference on Communication, Control and Computing, 1999.*

[2] Ravid Schwartz-Ziv and Naftali Tishby, Opening the black box of Deep Neural Networks via Information. *unpublished yet, available at* *https://arxiv.org/abs/1703.00810*

[3] Andrew Michael Saxe, Yamini Bansal, Joel Dapello, Madhu Advani, Artemy Kolchinsky, Brendan Daniel Tracey, David Daniel Cox, On the Information Bottleneck Theory of Deep Learning. *In Internation Conference on Learning Representations, 2018,* *https://openreview.net/forum?id=ry_WPG-A-*

[4] Sergey Ioffe, Christian Szegedy, Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *In Proceedings of the 32nd International Conference on Machine Learning, PMLR 37:448-456.*