REVISITING THE INFORMATION PLANE

Anonymous authors
Paper under double-blind review

ABSTRACT

There has recently been a heated debate (e.g. Schwartz-Ziv & Tishby (2017), Saxe et al. (2018), Noshad et al. (2018), Goldfeld et al. (2018)) about measuring the information flow in Deep Neural Networks using techniques from information theory. It is claimed that Deep Neural Networks in general have good generalization capabilities since they not only learn how to map from an input to an output but also how to compress information about the training data input (Schwartz-Ziv & Tishby, 2017). That is, they abstract the input information and strip down any unnecessary or over-specific information. If so, the message compression method, *Information Bottleneck* (IB), could be used as a natural comparator for network performance, since this method gives an optimal information compression boundary. This claim was then later denounced as well as reaffirmed (e.g. Saxe et al. (2018), Achille et al. (2017), Noshad et al. (2018)), as the employed method of mutual information measuring is not actually measuring information but clustering of the internal layer representations (Goldfeld et al. (2018)). In this paper, we will present a detailed explanation of the development in the Information Plain (IP), which is a plot-type that compares mutual information to judge compression (Schwartz-Ziv & Tishby (2017)), when noise is retroactively added (using binning estimation). We also explain why different activation functions show different trajectories on the IP. Further, we have looked into the effect of clustering on the network loss through early and perfect stopping using the Information Plane and how clustering can be used to help network pruning.

1 INTRODUCTION

Deep Neural Networks (DNNs) have recently achieved promising results in many areas especially computer vision and natural language processing. Yet, the learning process and design principles of configuring DNN architecture are under-investigated (Tishby & Zaslavsky, 2015). There are some recent attempts towards addressing this challenge. From an information theoretic viewpoint, Schwartz-Ziv & Tishby (2017) have investigated the learning dynamics of DNN – how the mutual information (MI) of the layer activation with input and target develops over the course of training. The finding is that DNNs generally first increase the MI of the layers with both, but then reduce the MI with the input. This perceived compression has led to promising results of DNN in many applications[1]. This compression behaviour resembles the IB-method, a constraint method which aims to retain maximum information content for given compression levels (Tishby et al. (1999)) and these possible maxima are depicted by the IB-bound.[2] Through the similarity, the IB-bound could be used as a way to judge network architecture (Schwartz-Ziv & Tishby (2017)). The closer to the IB-bound the better the NN is likely to perform. However, this finding is controversial, which has been supported by e.g. Achille et al. (2017); Achille & Soatto (2017); Noshad et al. (2018) and denied. Most prominently, Saxe et al. (2018) have argued that this does not generalize for all activation functions and that compression does not necessarily lead to good generalization.

Nonetheless Alemi et al. (2016), Kolchinsky et al. (2017), Nguyen & Choi (2018), Banerjee & Montúfar (2018) and Alemi et al. (2018) have tried to implement the IB-constraint as optimization parameter for DNN training leading to promising results. Amjad & Geiger (2018) criticize these attempt claiming that they were not really sticking to the IB for their optimization process since in deterministic NNs the mutual information is either infinite or constant. Hence, the IB cannot produce

[1] Good results are debatable see appendix F.

[2] see appendix E for more details.

optimizeable gradients. They therefore reason, that the results of these authors were only possible by giving up a hard IB constraint.

Recent success with fully invertible neural networks (which cannot experience any form of compression) cast doubt on the notion of compression being a necessary factor for good generalization (e.g. (Jacobsen et al., 2018), (Chang et al., 2017), (Ardizzone et al., 2019), (Song et al., 2019)).
Finally, a recent paper by Goldfeld et al. (2018) assessed that measuring MI in this scenario is actually tracking how clustered the internal layer representations of the samples are.

Building on Goldfeld et al. (2018), this work attempts to explain the trajectories in the IP created through MI estimation using binning. Through this we will shed more light on the investigation of the learning dynamics of DNNs through usage of the IP. Section 2.2.1 shows that the smaller the bin size for the binning estimator, the more the layers drift towards a fixed point in the IP. Section 2.2.2 highlights that higher layers strongly influence the shape of lower layers in the IP. Section 2.3 explains why the IP looks the way it does.[3] Clustering is then examined as a design parameter. This is done by investigating the connection between the loss function and clustering through the usage of early and perfect stopping in section 2.4.1. Here, no clear connection is found. Lastly, network pruning is attempted in section 2.5 using the IP, where slightly positive indications are found. At first though, the experimental setup is outlined.[4]

2 EXPERIMENTS

This section will introduce the experiments undertaken with the framework. Firstly, set goals are explained. Secondly, the experimental design is layed out. Lastly, the results are presented.

2.1 EXPERIMENTAL DESIGN

To make the experiments replicable and comparable, we use the same experiment setup of Schwartz-Ziv & Tishby (2017) and Saxe et al. (2018). To do so networks with the following designs have been chosen:

- 5 fully connected intermediate layers with 10-7-5-4-3 neurons;
- input and output layer size depend on the used dataset; that is, 12-2 for Schwartz-Ziv & Tishby (2017) and 784-Amount of Classes for MNIST;
- Softmax activation function for output as all experiments are classifications.

The activations of each layer are tracked for each iteration. Full tracking in beginning since the fastest movement is to be expected there and increased gaps for later iterations will help reduce computation complexity. Then the activations are binned. To judge the clustering of the activations, bin histograms are used for different layers. We also follow the same design by using and comparing networks with TanH and ReLU activations functions. Additionally combinations of the two are used to check for potential interference. Hence these 4 combinations:

- Input-TanH-TanH-TanH-TanH-TanH-Softmax
- Input-ReLU-ReLU-ReLU-ReLU-ReLU-Softmax
- Input-TanH-TanH-ReLU-ReLU-ReLU-Softmax
- Input-ReLU-ReLU-TanH-TanH-TanH-Softmax

The first one is the activation setup by Schwartz-Ziv & Tishby (2017) and the second one is by Saxe et al. (2018). All of the networks do not include any form of regularization and use SGD. Initially the experiments are run on the same dataset as in Schwartz-Ziv & Tishby (2017). Additionally, the MNIST-dataset (Lecun et al. (1998)) is used as in Saxe et al. (2018) to have a more complicated case with more features.[5]

[3] If needed, relevant information theoretic concepts are briefly explained in appendix A.

[4] Code can be found at `https://drive.google.com/open?id=1CwBthST4bwtvMtw39Fy1bWMQ6hrz0Hzo`

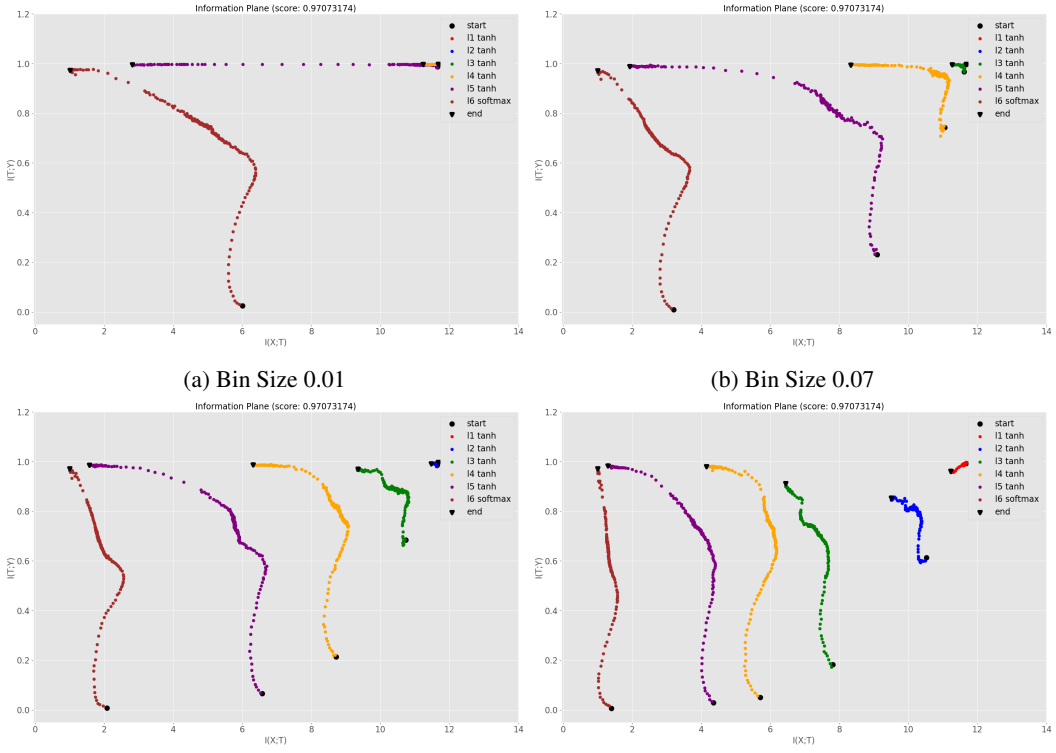
[5] To simplify the computation, a subset of MNIST is used. Results for MNIST can be seen in Appendix D.

The batch size is standardized to 256. To estimate mutual information binning is used as it allows to easily track the clustering. Different bin sizes are considered: 0.01, 0.07, 0.15 and 0.3. Schwartz-Ziv & Tishby (2017) have used the amount of bins, but to allow a better comparison between TanH- and ReLU-networks, it is chosen to change to use bin size instead since 20 bins for a given range of 2 and an undefined range are deemed less compare-able.
The size of 0.07 corresponds to the 30 bins chosen in Schwartz-Ziv & Tishby (2017); Saxe et al. (2018); that is, Distance $2\backslash 30 \approx 0.07$. 0.01 has been chosen as a smaller bin size because as stated in Cover & Thomas (2005) and Kraskov et al. (2004) smaller bin sizes make it more likely that the estimation error is small and more reliable results are achieved. The bigger bin sizes have been chosen to investigate the influence on a wider range of values. The resulting information planes are plotted in a different way than in prior works. The idea is to plot the IP with different colours for the different layers to make them distinguishable. After that the information planes for early stopping and perfect stopping (stop at minimum validation loss) are investigated. Lastly, we have looked into whether an early stopping IP of a bigger network gives indications if layers can be pruned.

2.2 Results

2.2.1 Influence of Bin Size

From Figures 1a, 1b, 1c and 1d in that order, one interesting find is that for TanH-networks depending on the bin size more layers drift towards the top right when the bin size decreases (therefore the amount of bins increases). The effect is stronger the higher level the layer is. Also, the output layers end point is not affected by that. Thus, the bin size is clearly having a big effect on the IP.



(a) Bin Size 0.01

(b) Bin Size 0.07

(c) Bin Size 0.15

(d) Bin Size 0.3

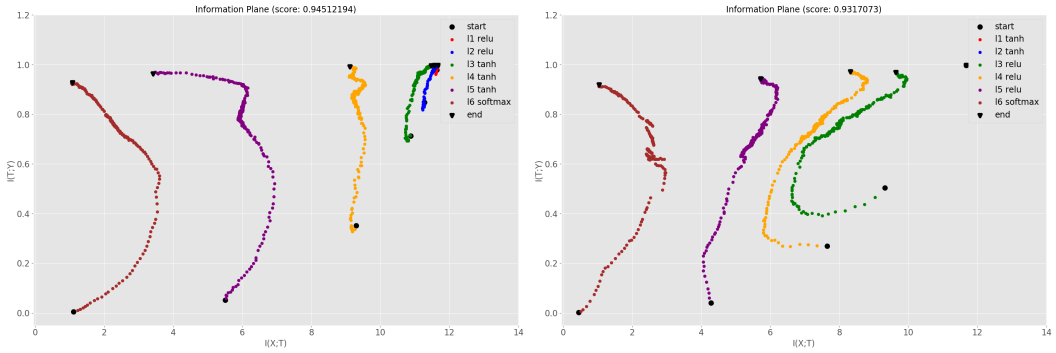
Figure 1: Information planes for different bin sizes for with batch size 256 (Schwartz-Ziv & Tishby (2017)-dataset)

2.2.2 Mixed Networks

Looking at the results of the mixed networks, one can see that ReLU-layers influence deeper TanH-layers. Figure 2a shows the IP of a network where the higher layers are ReLU and the lower TanH.

One can see that the layers following the ReLU-layers changed their pathway to a similar shape than the ReLU-layer itself and not the "normal" pathway of 2 compression phases. This effect gets weaker the deeper in the network. Figure 2b shows the IP for a network with TanH in higher and ReLU in lower layers. Here there is a significant impact of the TanH-layers on the ReLU-layers findable. The ReLU-layers take on a similar behaviour to the TanH-layers even though the TanH-layers show no compression at all. The effect gets also weaker the further the layer is apart from the TanH effect.

These effects have not always been present or prominent for every time mixture networks have been trained (see fig. 9 in appendix B and fig. 12a in appendix D). Nonetheless, there seems to be an influence of the previous layers on the compression in the lower layers.



(a) Leading ReLU

(b) Leading TanH

Figure 2: Information plane for Mix-Network for batch size 256 and bin size 0.07 (Schwartz-Ziv & Tishby (2017) dataset)

2.3 The Effect of the Introduction of Noise through Binning on the Information Plane

According to Goldfeld et al. (2018); Saxe et al. (2018); Amjad & Geiger (2018), in the TanH-networks, mutual mutual information is either infinite (for continuous input features) or constant (for discrete input features). Because the datasets in usage here are both discrete, the mutual information has to be a constant; that is, MI of the input with the output $I(X;Y)$ has to be the same as the MI of the layers with the output $I(T_i;Y)$. Also given the input X $I(T|X)$, the conditional entropy of the layer activities T is 0, which is explained in Appendix A.1. Therefore, we calculate these values for comparison and Table 1 shows the results for the binning estimator[6].

Comparing these values to the top right points in figures 1 and 11 (MNIST) one can see that these values seem to exactly represent the most extreme point of the information plane on the top right where the layers drift towards if the bin size is reduced ($H(X)$ as max $I(X;T)$ and $I(X;Y)$ as max $I(T;Y)$). Hence, this gives evidence, that the mutual information is as claimed constant and that conditional entropy $I(T|X) = 0$ which results in $I(X;T) = H(X)$.

Dataset	I(X;Y)	H(X)
Schwartz-Ziv & Tishby (2017)	0.9976734295143714	11.677719641641012
MNIST	1.5849625007211579	10.550746785383243

Table 1: Mutual information between input and output and entropy of input for Schwartz-Ziv & Tishby (2017)- and MNIST-dataset

However, this leads to the question: *why does the information plane not show constant MI?*. According to Saxe et al. (2018), using a binning method, one loses the invertibility of the TanH function and a bin almost by definition loses information about the value put in the bin (e.g. if one puts 2.53 and

[6] $H(X)$ is simply $\log_2(N)$ where N is the number of samples since each sample is different and each have a probability of $p(x_i) = \frac{1}{N}$. $\log_2(x) \geqslant 0$ for $x \geqslant 1$.

2.54 in the 2.55 bin one cannot trace which one is which afterwards). The smaller the bin size and therefore, the more bins there are, the better is the estimation by default (see appendix G). Section 2.2.1 establishes that the more bins the estimator uses the more of the layers drift towards the top right which indicates that mutual information is indeed constant.

Referring to Appendix A.1, mutual information of the input X with a layer T is defined as $I(X;T) = H(X) - H(X|T)$. Since $H(X)$ is a constant, a change in mutual information with the layer has to be a result of a change in $H(X|T)$. Because $H(X|T)$ is only 0 if X is a function of T, one can infer that if there is no direct mapping between X and T, there will be an increase in $H(X|T)$ which will reduce the mutual information between the two. The same applies to the output Y.

Figure 3 shows different sample cases of how one can lose mutual information through binning. In the first case there are two samples from two different classes placed into one bin. This makes it impossible to differentiate between classes given only the bin number. This leads to a decrease in mutual information of the layer with the output since there is no direct mapping between the array which contains this bin and an output class possible (see Appendix G where array building is explained). Also, since there are two samples in one bin, one cannot differentiate between the two samples and therefore one loses mutual information with the input of the network since it is not possible to directly map between the input and the bin. The second and third case have two samples of the same class. Hence, it will only lead to a decrease in $I(X;T)$.

For example, two samples from two different classes are producing the activations 0.1, 0.33, 0.4 and 0.1, 0.34, 0.4. If no binning was performed, this could lead to the arrays 0103304 and 0103404 (this is just an explanatory example in reality they get transformed to binary first). Each sample is uniquely identifiable and each sample can get a class allocated. However, if there was binning in 0.05 steps one would end up with the two arrays 0103504 and 0103504 and no differentiation is possible.

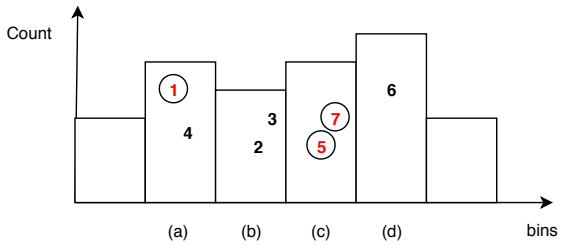


Figure 3: Example cases of how to lose mutual information. Different colours (and circled) represent different classes and different numbers are different samples (a) two different classes in one bin $\rightarrow$ decrease I(T;Y) and I(X;T) (b) two samples of same class in one bin $\rightarrow$ decrease I(X;T) (c) two samples of the other class in one bin $\rightarrow$ decrease I(X;T) (d) single sample in a bin $\rightarrow$ no information loss

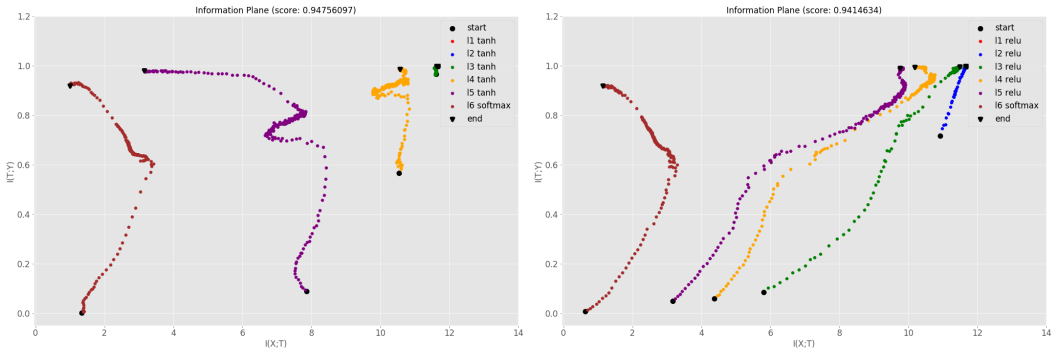
If one plots the bins in a histogram at the beginning of training, at half of the epochs and after training (see fig. 5) and compares it with the movement in the information plane (see fig. 4), one can see that this roughly seems to fit this notion. As the layers which show compression with respect to the input also show that their amount of empty bins increases over the course of the training and their clustering into smaller amounts of bins increases.

So why do the bins empty out? This is driven by the Softmax output-layer whose neurons during training are going to be driven to output either 1 or 0 for each sample to learn to correctly classify. Hence, by definition this layer is going to reduce its filled bins down to two bins - 0 and 1. This explains where the output layer is drifting towards $I(X;T) = 1$ since the entropy will be 1 and there is only 1 bit of information needed to find out which state the activation is in.

Figure 5 shows that the deeper the layer, the more the activations get binned into the extreme bins for TanH and the more training proceeds the more bins are emptied out. The output layer seems to force the next higher layer to take on extreme values as well as itself gets more and more decisive on the sample classes. This gets propagated to the next layer and so on, but loses intensity. Figure 4

demonstrates that this matches the corresponding information plane. The two layers that get clustered show compression while the others do not.[7]

Therefore, one can derive that a concentration of the activations into a smaller amount of bins results in lower estimated mutual information with the input.



(a) TanH-Network IP (Bin size 0.07)

(b) ReLU-Network(Bin size 0.07)

Figure 4: Information planes for TanH and ReLU where bins were tracked. TanH is different than in fig. 2b because it is from a different notebook that did not gather all data at the same time.

The situation with the output Y is different, since information is only lost when there are samples of different classes in one bin leading to equal arrays (see above example). In the beginning we have very low mutual information of the layers and the output. This is because we have random weight initialization and the samples are more or less randomly allocated to the bins. This will cause a lot of bins to have samples of different classes. As training goes on, the network learns to better differentiate between classes and will be more likely to put samples from the same classes in the same binning areas. The best example is the output layer where the network either produces 0 or 1 at the end of training. Since Softmax is used this will produce an array of multiple zeros (depending on the number of classes -1) and one 1 (see appendix G: activations get transformed into continuous array for binning). This allows to uniquely identify which class the sample is allocated to by the network.

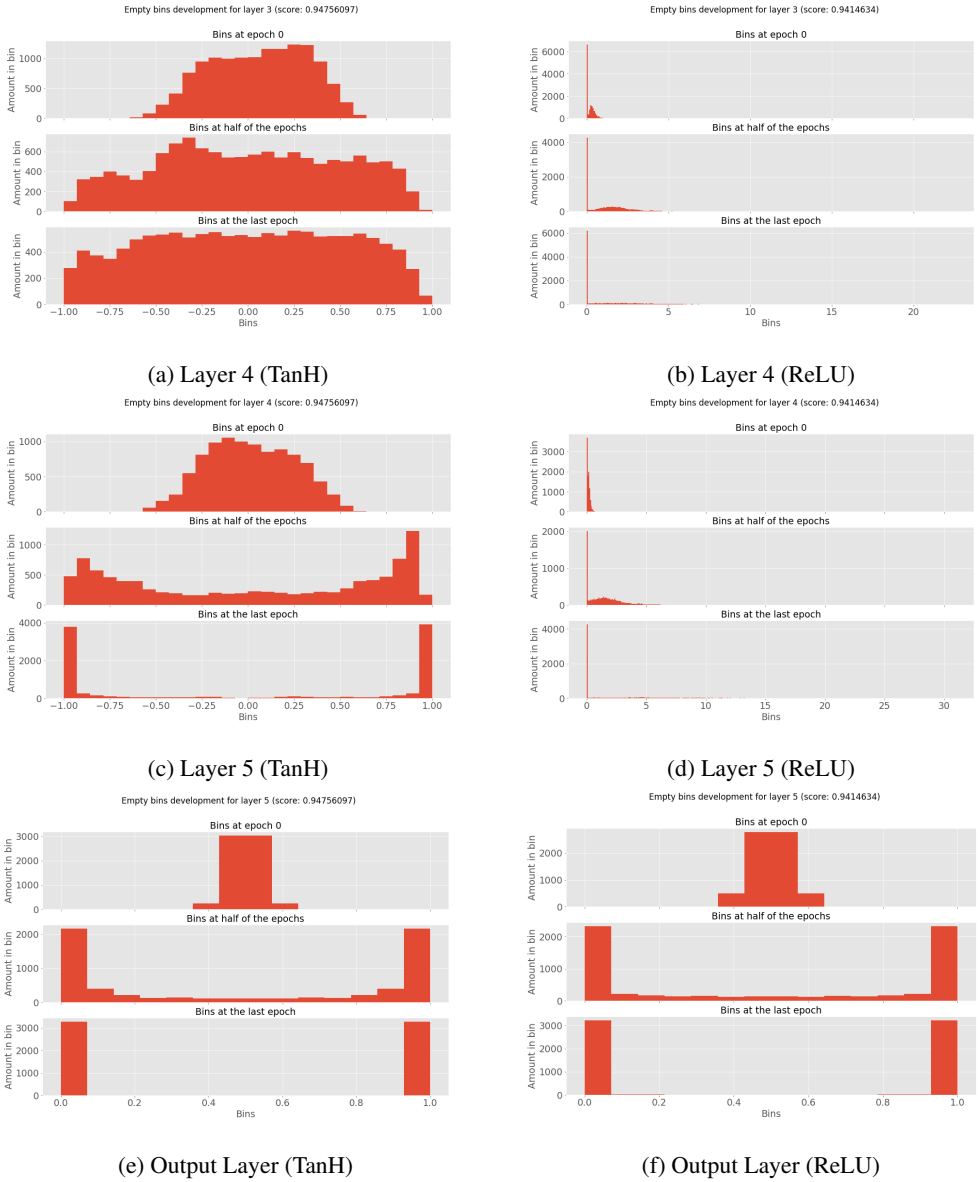
The next question about the pathway of the layers in the information plane is: *why do ReLU-layers take such a different path than TanH-layers?*. The reason is that ReLUs has an inbuilt bin, because activations can only be non negative. The weights of the network get initialized randomly and often are negative values. This make the input of the ReLU activation function negative and because of that the result will be 0. Hence, a lot of the activations are going to be 0 in the beginning. This means that the total range of our bins will have less activations available to fill bins and for the same argument as before this will lead to a lower entropy which will lead to less estimated mutual information. Over the course of training more and more of the activation functions are leaving the 0-bin and filling previously empty bins and this ”frees” information which results in the ReLU-layers path towards the top right. This can be observed in the plots 5b, 5d and 5f in Figure 5 together with the IP in Figure 4b where there is a wider spread of activations in bins. As training goes on, the amount of empty bins is decreasing for the ReLU-layers. Thus entropy is increasing, which in return increases the estimated mutual information.

Figures 5e and 5f also show why the the output layer initially drifts to an increase in MI with the input. It is because there is an initial drop in empty bins since in the beginning the spread in the bins is very small (big aggregation in the center at the beginning). Figures 14-13 presents the same trend for the MNIST-dataset[8].

The remaining question is why a larger bin size results in the layers drifting away from the “real” mutual information. Naturally, larger bins result in fewer total bins which makes it more likely that

[7]It is important to note though that empty bins are just an indicator. It could be that there are few empty bins but some with large sample concentration for the same result.

[8]Here the bin size has been increased since a too small bin size results in very thin bins that are not visible.



(a) Layer 4 (TanH)

(b) Layer 4 (ReLU)

(c) Layer 5 (TanH)

(d) Layer 5 (ReLU)

(e) Output Layer (TanH)

(f) Output Layer (ReLU)

Figure 5: Bin histograms for TanH and ReLU network at epochs 1, 4000 and 8000 for Schwartz-Ziv & Tishby (2017) dataset.

multiple activations land in the same bin. Thus, it is more likely that the built array is the same which in returns leads to information loss.

2.4 BOTTLENECKING

Another way to show that clustering is at play is if one inserts a bottleneck into the NN. Figure 6 shows the IP for a network with a 12-3-2-12-2-2 layer-neuron composition. The 3rd layer which is the bottleneck actually shows higher compression than the 4th. If one considers data processing inequality this should be impossible, but it is purely a result of many samples being in the same bins.

In summary, there seems to be a strong indication that the information plane is merely tracking clustering as suggested by Goldfeld et al. (2018). Hence, the advantages of clustering should be investigated. This is done in the following section.

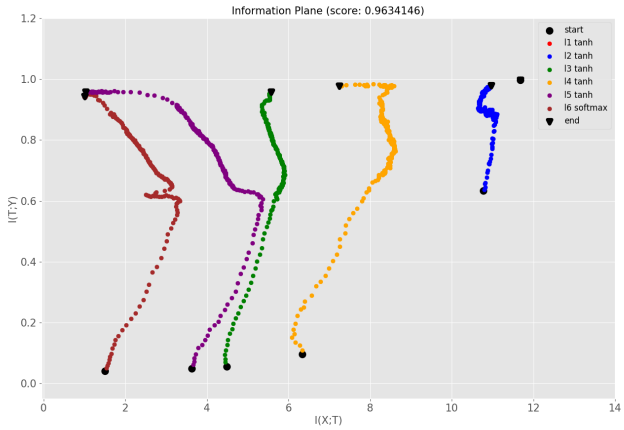


Figure 6: IP for Bottleneck network for Schwartz-Ziv & Tishby (2017) dataset.

2.4.1 Early and Perfect Stopping

Since it does not seem that the highest level of compression is beneficial for generalization (see appendix F or (Saxe et al., 2018), it makes sense to see what happens when one uses a fairly standard way of trying to achieve good generalization - early stopping.

Figure 7a shows the results of early stopping and fig. 7b for perfect stopping compared to the full 8000 iterations for a TanH-network[9]. It seems that early stopping occurs when the network is starting to enter the compression phase of the output-layer (early stop point is in the curve turning point). The same happens for a ReLU network (see fig. 10a). This stopping does not generalize to the MNIST-dataset (see Figure 15 in Appendix D).

Figure 7b presents that early stopping does not guarantee to stop at the global minimum, as it is decided to see what the information plane shows at the global minimum for a "perfect" stop. Therefore, there seems to be a not general connection between compression and the first minimum but no connection to the global minimum.

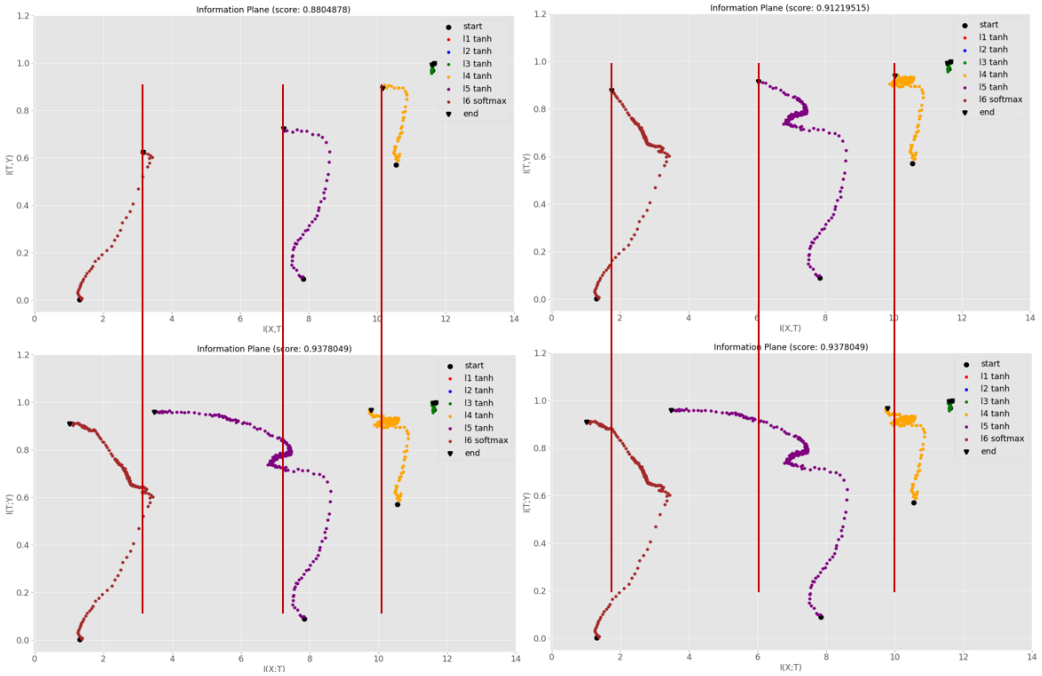
2.5 Clustering for pruning

We believe, that in deep neural networks, layers at the end of the network are more susceptible to be pushed into a clustered state by the output layer when they are less needed by the network to map the input to the output, since the network is essentially just passing the information through without refining it. It has been shown that sometimes much smaller networks can perform similar generalization performance than bigger ones (e.g. le Cun et al. (1990), Giles & Omlin (1994), Srinivas & Babu (2015), Zhang et al. (2017)). Hence, clustering could potentially be used to identify less effective layers at the end which could be safely removed to reduce computational complexity.

To investigate this the amount of hidden layers is doubled by adding additional 3 neuron layers at the end, assuming that these are potentially prune-able. Additionally early stopping is performed to have a more realistic scenario. The resulting IPs[10] can be seen in Figure 8. The goal is to remove layers that have similar trajectories from the network; e.g. layers 5,6 and 7 in Figure 8a. This process is repeated until there are no similar layers and for each pruning, we re-plot the IP. Table 2 presents the scores after each pruning. Both network types have shown prune-able layers until all added layers have been removed. All pruning actions does not result in decreased score. We are aware that this is a very much designed scenario, but it shows that this clustering-based pruning might be worthy of further investigation.

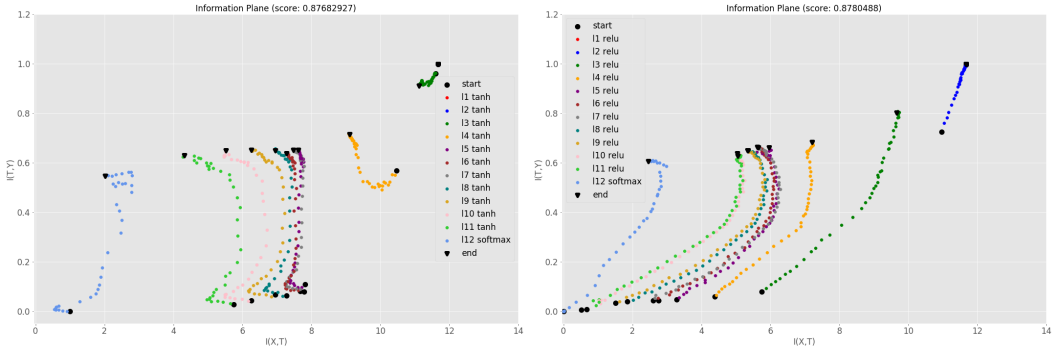
[9]Results for ReLU network in fig. 10 in appendix C

[10]Because running time is limited, here only Schwartz-Ziv & Tishby (2017) dataset is used.



(a) TanH (b) TanH

Figure 7: Comparison of information planes for early (left) and perfect (right) stopping (top) and full iterations (bottom) of TanH network on Schwartz-Ziv & Tishby (2017)-dataset with batch size 256 and bin size 0.07



(a) Big TanH-Network IP (b) Big ReLU-Network

Figure 8: Big TanH and ReLU-networks on Schwartz-Ziv & Tishby (2017)-dataset

3 DISCUSSION

It has been reaffirmed that the information plane only seems to track clustering of the activations, and now we will revisit and summarise the previous findings. We have highlighted the effect of mixing activation functions. Adding ReLU-layers before TanH-layers changes the behaviour on the TanH-layers, which is often observed on ReLu-layer. With clustering in mind, the reason can be that in the beginning of training ReLU-layers often have 0 as activation[11] which is going to be transmitted to the TanH neurons. No matter what the weight of the connection is, TanH is going to receive a 0. This will lead to a clustering of the TanH-activations in the 0-bin which will lead to an initial reduced mutual information with the input and the output. The longer training goes the more will the

[11]Random weight initialization leading to negative weights.

Network type	Big	1st Prunning	2nd	3rd
TanH	0.8768	0.8780	0.8732	0.8805
ReLU	0.8780	0.8646	0.8805	

Table 2: Accuracy scores for big and prunned networks for Schwartz-Ziv & Tishby (2017)-dataset

ReLU-layer usually diversify the activations which in return will also lead to a diversification in the TanH activations. Hence $I(X;T)$ and $I(T;Y)$ will increase.

When there is a TanH-layer in front of a ReLU-layer, in the beginning there will be many negative activations sent to the ReLU-layers since TanH ranges from -1 to 1 and this will force more activations of the ReLU-layer to be 0. Random weight initialization will sometimes make this effect stronger or weaker (see Appendix B). This will lead to an initial decrease in the MI before the network stabilizes and learns to adjust the weights accordingly. Therefore, the composition of the network will influence the clustering in the network.

On the one hand there seems to be a connection between clustering in the output-layer and Early Stopping, on the other hand this does not apply to global minima. The results on the MNIST dataset have shown that this connection does not generalize for different datasets. This is another indication that the claims of Schwartz-Ziv & Tishby (2017) do not generally hold even when translated from compression to clustering. Nonetheless, the pruning experiments have shown that clustering might be useful for NN design.

4 Conclusion

This paper studies the information plane as a neural network analysis tool. We have looked into the influence of different bin sizes for binning estimation, which has led to a detailed explanation of why certain behaviour is happening in the information plane. Thus, finding new strong evidence that the information plane only tracks clustering as Goldfeld et al. (2018) suggested. The usage of measuring clustering has been investigated using early stopping and perfect stopping, which we have not been able to generalise the finding across different datasets. Clustering can be used to design a NN in terms of pruning, which might be worthy of further investigation. The information plane holds value as a measure of clustering and could potentially lead to advancements in Deep Learning.

One aspect that has not been part of the discussion so far is that in contrast to non-linearly saturating activation functions like TanH, which has no binning during the real training process, ReLU in fact has a bin. The 0-bin could actually lead to a loss in mutual information because the injectiveness of the activation function gets lost (not invertible anymore) and mutual information is not bound to be constant or infinite. Therefore, networks with ReLU could experience a form of compression. ReLU does in general show better generalization capabilities than TanH, which could partially support the claim that compressed neural networks generalize better Schwartz-Ziv & Tishby (2017). A well known problem of ReLUs is called “dying ReLUS” which could be a case of “too high” compression. Which would disturb the mapping between input and output.

Taking out the binning of ReLUs, like in LeakyReLUs, is almost always favorable compared to standard ReLUs in terms of generalization (Xu et al. (2015)). Since LeakyReLUs restore the invertibility of the activation function and therefore prevent compression, this also indicates that compression does not necessarily generalizes better in DNNs. It remains a task for future investigations, how this can be explained in detail.

References

Alessandro Achille and Stefano Soatto. Emergence of Invariance and Disentanglement in Deep Representations. jun 2017. URL http://arxiv.org/abs/1706.01350.

Alessandro Achille, Matteo Rovere, and Stefano Soatto. Critical Learning Periods in Deep Neural Networks. nov 2017. URL http://arxiv.org/abs/1711.08856.

Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. Deep Variational Information Bottleneck. dec 2016. URL http://arxiv.org/abs/1612.00410.

Alexander A. Alemi, Ian Fischer, and Joshua V. Dillon. Uncertainty in the Variational Information Bottleneck. jul 2018. URL http://arxiv.org/abs/1807.00906.

Rana Ali Amjad and Bernhard C. Geiger. Learning Representations for Neural Network-Based Classification Using the Information Bottleneck Principle. feb 2018. doi: 10.1109/TPAMI.2019.2909031. URL http://arxiv.org/abs/1802.09766http://dx.doi.org/10.1109/TPAMI.2019.2909031.

Lynton Ardizzone, Carsten Lüth, Jakob Kruse, Carsten Rother, and Ullrich Köthe. Guided Image Generation with Conditional Invertible Neural Networks. jul 2019. URL http://arxiv.org/abs/1907.02392.

Pradeep Kr. Banerjee and Guido Montúfar. The Variational Deficiency Bottleneck. oct 2018. URL http://arxiv.org/abs/1810.11677.

Bo Chang, Lili Meng, Eldad Haber, Lars Ruthotto, David Begert, and Elliot Holtham. Reversible Architectures for Arbitrarily Deep Residual Neural Networks. sep 2017. URL http://arxiv.org/abs/1709.03698.

R Clausius. *The Mechanical theory of heat with its applications to the steam-engine and to the physical properties of bodies*. 1867.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. 2005. ISBN 9780471241959. doi: 10.1002/047174882X.

C.L. Giles and C.W. Omlin. Pruning recurrent neural networks for improved generalization performance. *IEEE Transactions on Neural Networks*, 5(5):848–851, 1994. ISSN 10459227. doi: 10.1109/72.317740. URL http://ieeexplore.ieee.org/document/317740/.

Ziv Goldfeld, Ewout van den Berg, Kristjan Greenewald, Igor Melnyk, Nam Nguyen, Brian Kingsbury, and Yury Polyanskiy. Estimating Information Flow in Deep Neural Networks. oct 2018. URL http://arxiv.org/abs/1810.05728.

Jörn-Henrik Jacobsen, Arnold Smeulders, and Edouard Oyallon. i-RevNet: Deep Invertible Networks. feb 2018. URL http://arxiv.org/abs/1802.07088.

Artemy Kolchinsky, Brendan D. Tracey, and David H. Wolpert. Nonlinear Information Bottleneck. may 2017. URL http://arxiv.org/abs/1705.02436.

Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical Review E*, 69(6):066138, jun 2004. ISSN 1539-3755. doi: 10.1103/PhysRevE.69.066138. URL https://link.aps.org/doi/10.1103/PhysRevE.69.066138http://arxiv.org/abs/cond-mat/0305641http://dx.doi.org/10.1103/PhysRevE.69.066138.

Yann le Cun, John S. Denker, and Sara A. Solla. Optimal Brain Damage. In David S. Touretzki (ed.), *Advances in neural information processing systems*, chapter Optimal Br, pp. 598–605. Morgan Kaufmann Publishers Inc., 1990.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. ISSN 00189219. doi: 10.1109/5.726791. URL http://ieeexplore.ieee.org/document/726791/.

George Markowsky. Information theory, 2017. URL https://www.britannica.com/science/information-theory.

Thanh T. Nguyen and Jaesik Choi. Parametric Information Bottleneck to Optimize Stochastic Neural Networks. In *ICLR 2018 Conference Blind Submissions*, 2018. URL https://openreview.net/pdf?id=ByED-X-0W.

Morteza Noshad, Yu Zeng, and Alfred O. Hero. Scalable Mutual Information Estimation using Dependence Graphs. jan 2018. URL http://arxiv.org/abs/1801.09125.

Andrew Michael Saxe, Yamini Bansal, Joel Dapello, Madhu Advani, Artemy Kolchinsky, Brendan Daniel Tracey, and David Daniel Cox. On the Information Bottleneck Theory of Deep Learning. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=ry{_}WPG-A-.

Ravid Schwartz-Ziv and Naftali Tishby. Opening the Black Box of Deep Neural Networks via Information. mar 2017. URL http://arxiv.org/abs/1703.00810.

Claude. E. Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal*, 27 (3):379–423, jul 1948. ISSN 00058580. doi: 10.1002/j.1538-7305.1948.tb01338.x. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6773024.

Yang Song, Chenlin Meng, and Stefano Ermon. MintNet: Building Invertible Neural Networks with Masked Convolutions. jul 2019. URL http://arxiv.org/abs/1907.07945.

Suraj Srinivas and R. Venkatesh Babu. Data-free parameter pruning for Deep Neural Networks. jul 2015. URL http://arxiv.org/abs/1507.06149.

Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop, ITW 2015*, 2015. ISBN 9781479955268. doi: 10.1109/ITW.2015.7133169.

Naftali Tishby, Fernando C. Pereira, and William Bialek. The Information Bottleneck Method. *The 37th annual Allerton Conference on Communication, Control, and Computing*, pp. 368–377, 1999.

Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical Evaluation of Rectified Activations in Convolutional Network. may 2015. URL http://arxiv.org/abs/1505.00853.

Zhi Zhang, Guanghan Ning, and Zhihai He. Knowledge Projection for Deep Neural Networks. oct 2017. URL http://arxiv.org/abs/1710.09505.

A BACKGROUND

The following addresses some necessary aspects of information theory to then briefly introduce the IB method and excerpts of the debate about its applicability in deep learning. Information theory is a mathematical way to represent the transmission and processing of information (Markowsky (2017)). It is mostly founded on work of Claude Shannon (Shannon (1948)). There lies interest in studying information theory when trying to understand neural networks, as in a neural networks information of an input X about an output Y gets propagated through the network in an attempt to learn the mapping of this information and with this to then generalize this mapping on unseen data. For the later, two important concepts "entropy" and "mutual information" have to be understood which are briefly outlined in the following. After that the IB method is summarised.

A.1 ENTROPY AND MUTUAL INFORMATION

Entropy is a term coined by Clausius (1867). It originates from thermodynamics and is a measure of disorder in a system or how uncertain events are. It denotes the average amount of information needed to describe a random variable. For a discrete random variable X, it can be expressed the following way:

$$H(p) = -\sum_{x \in X} p(x) log_2(p(x)) \tag{1}$$

where $p(x)$ is the probability mass function and the result is denoted in information bits.
Conditional entropy is the entropy of a random variable conditioned on another random variable's knowledge (Cover & Thomas (2005)). Between two distributions X and Y it is calculated like this:

$$\begin{aligned} H(Y|X) &= \sum_{x \in X} p(x) H(Y|X=x) = -\sum_{x \in X} p(x) \sum_{y \in Y} p(y|x) \log p(y|x) \\ &= -\sum_{x \in X} \sum_{y \in Y} p(x,y) \log p(y|x) \end{aligned} \tag{2}$$

an important property of conditional entropy is that the conditional entropy of X given Y $H(X|Y)$ is 0 is and only if X is a deterministic function of Y (Cover & Thomas (2005)).
Relative entropy also known as Kullback-Leiber divergence is a measure of instance between the two distributions. It can also be interpreted as how inefficient it is to assume the probability distribution q if the real one is p. If one falsely assumes q the average description length would change to $H(p) + D(p||q)$ which is the sum of the entropy and the Kullback-Leiber divergence. Kullback-Leiber divergence is defined as follows:

$$D(p||q) = \sum_{x \in X} p(x) \log \left(\frac{p(x)}{q(x)} \right) \tag{3}$$

this always non negative and only 0 if $p = q$ (Cover & Thomas (2005)).
The other relevant metric which is Mutual Information (MI). MI measures how much information one variable has about another one. It is therefore a measure of dependency and in contrast to the linear correlation coefficient, it also measures dependencies that do not show in covariance. An important property is that it is invariant to reparametizations. Having variables X and Y this means, that if there is a direct invertible mapping between variable X and a reparametized form of it X' [12], then $I(X;Y) = I(X';Y)$ (Kraskov et al. (2004)). MI can be calculated as follows (Cover & Thomas (2005), Schwartz-Ziv & Tishby (2017)):

$$\begin{aligned} I(Y;X) &= D_{KL}[p(x,y)||p(x)p(y)] = \sum_{x \in X, y \in Y} p(x,y) \log \left(\frac{p(x,y)}{p(x)p(y)} \right) \\ &= \sum_{x \in X, y \in Y} p(x,y) \log \left(\frac{p(x|y)}{p(x)} \right) = H(X) - H(X|Y) \end{aligned} \tag{4}$$

[12] X and X' are homeomorphisms

Mutual information can also be used as a measure of independence between two variables, as it is 0, only if the two variables are strictly independent (Kraskov et al. (2004)).

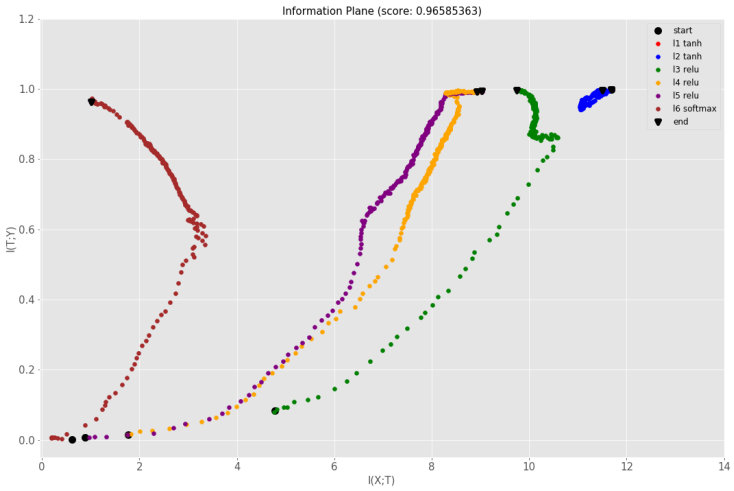
Information Bottleneck Method

Shannons information theory focused on transmission of information in signal processing. Signal transmission, is usually subject to some form of loss as not 100% of the information reach their recipient. This principle is known as the Data Processing Inequality (DPI), which states that the information content of some input cannot be increased through data manipulation (e.g. by reparameterization) (Cover & Thomas (2005)). It is possible, that one does not need all of the information to reconstruct the initial message. To do so, the relevant part of the information has to reach the recipient. This means, that the original message can be compressed until it only conveys this relevant part, which allows a direct mapping back to the original message (it is invertible).
Sometimes, one wants to compress information since information transfer is expensive.[13] The standard way to determine how compressed a message is, is to calculate mutual information $I(\widetilde{X}; X)$ where $\widetilde{X}$ denotes the compressed message and X the original (see eq. 4). The standard way to express how compressed a message can be is rate distortion theory which can be used to derive the bottom optimal distortion for a given distortion rate. This smallest possible representation is called a "minimal sufficient statistic". It allows to map the output to the input without mistakes (Cover & Thomas (2005)).
Tishby et al. (1999) introduced a way to not only compress the message, but also to compress it such that it retains a maximum of its information about an output Y measured through the mutual information of the input with the output $I(X; Y)$. Therefore, $\widetilde{X}$ becomes a minimal sufficient statistic of X with respect to Y. This means that it is the least complex mapping of X capturing the mutual information $I(X; Y)$ (Tishby & Zaslavsky (2015)). This method is called the IB method. The main advantage is, that this optimization problem has an exact solution the Information Bound in the IP (Tishby et al. (1999)).[14]
Tishby & Zaslavsky (2015) asserted that deep neural networks can be interpreted as Markov-Chains[15], since each layer is just depending on the previous layer's output. Thus, DPI would apply and each additional layer (ignoring single neurons) would have at most equal or less (mutual-)information about input and output than the previous one. They therefore assume that a DNN compresses relevant information much like the IB method down to something close to a minimal sufficient statistic. Through this, they claim that the IB limit, the Information Bound, serves as a natural comparator of DNN architecture performance - the closer to the IB Bound the more optimal the architecture.

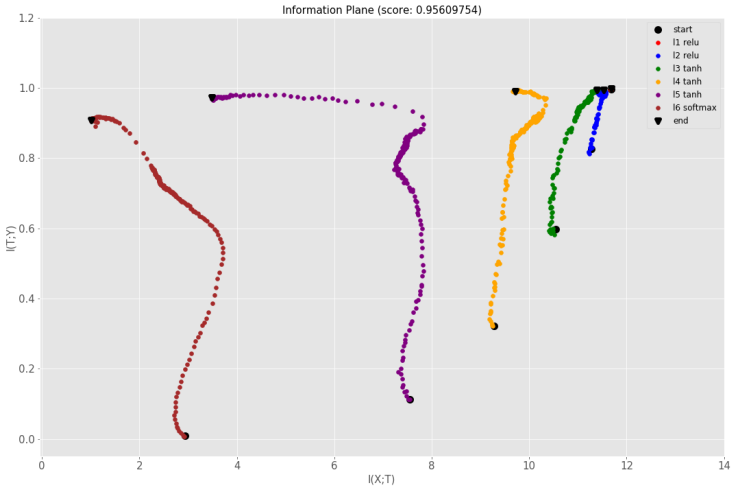
[13] E. g. sending a JPEG image is a lot faster and cheaper than sending a raw file.
[14] See ap. E fig. 16 for examples.
[15] A series of memory-less stochastic events.

B OTHER MIX NETWORK EXAMPLES

These examples show that the trajectory of the information planes is very susceptible to how the weights are initialised.



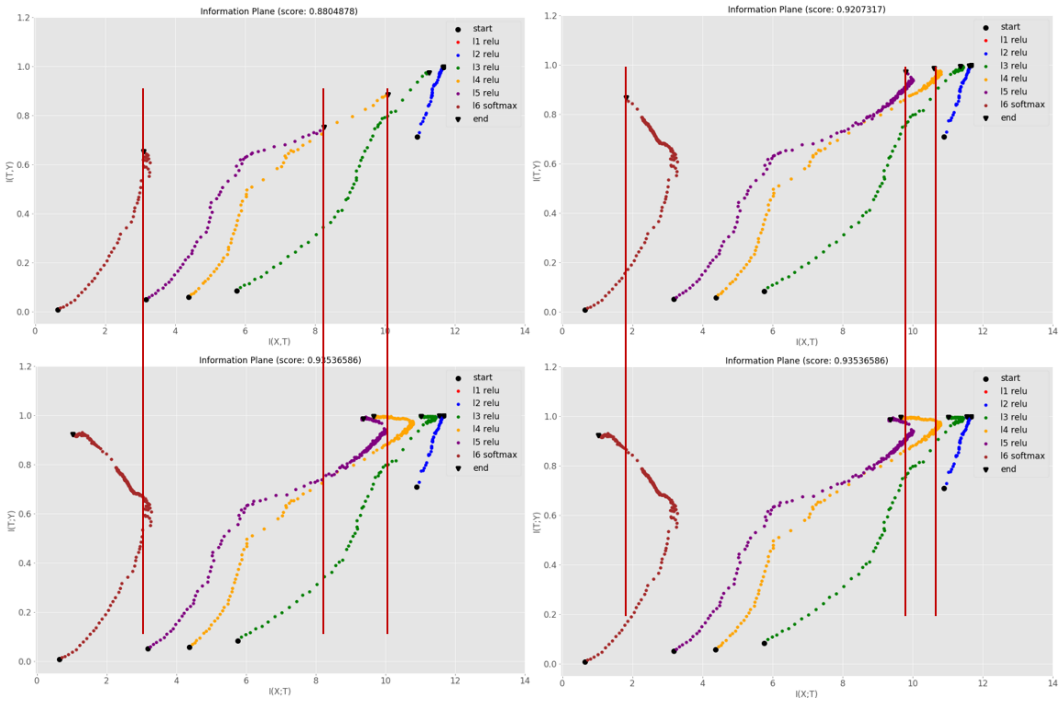
(a) Leading ReLU



(b) Leading TanH

Figure 9: Other information planes for mix networks for batch size 256 and bin size 0.07 (Schwartz-Ziv & Tishby (2017) dataset)

C ADDITIONAL RELU EARLY AND PERFECT STOP PLOTS



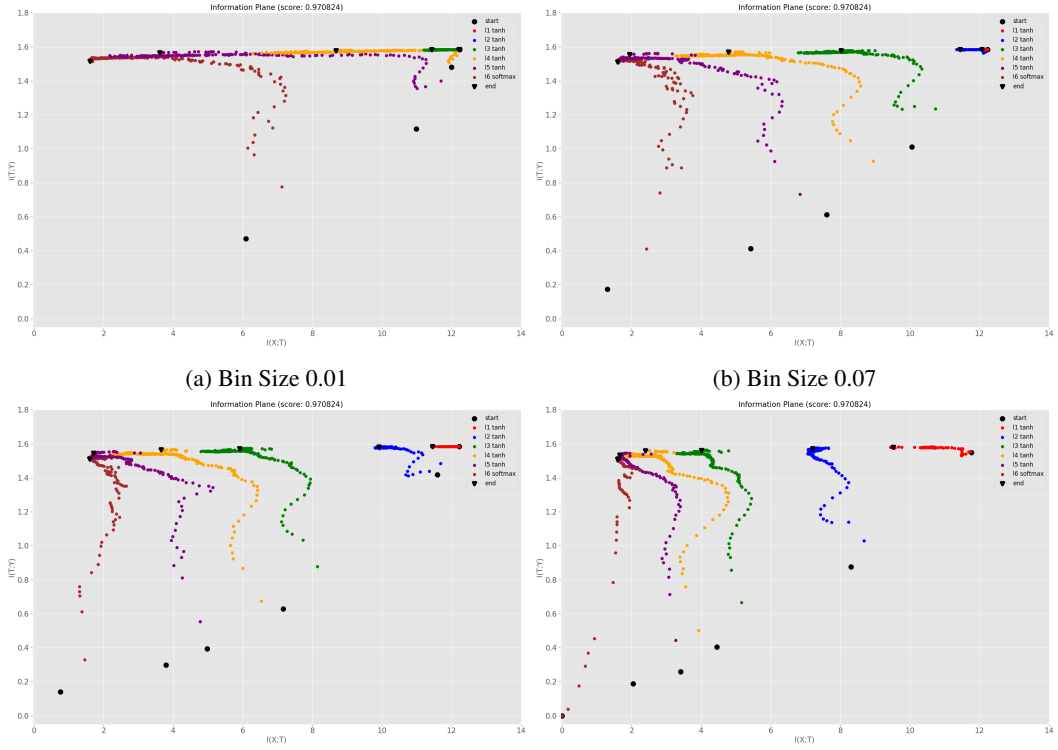
(a) ReLU (b) ReLU

Figure 10: Comparison of information planes for early (left) and perfect (right) stopping (top) and full iterations (bottom) of ReLU-network on Schwartz-Ziv & Tishby (2017)-dataset with batch size 256 and bin size 0.07

D MNIST-RESULTS

Concerning the results for MNIST using binning (see figure 11), one can see, that the general trend of the layer movement in the IP is the same as it was found for the Schwartz-Ziv & Tishby (2017)-dataset.[16] Another difference is that the mixture-networks are less susceptible to the different activation functions in the higher layers. There is no noticeable influence iff ReLU is in the higher layers (see the plots in figure 12a). For leading TanH-layers there is still a significant influence perceive-able in the early periods at the bottom for the green and the yellow layer (see plots in fig. 12b).
The biggest difference is, that there is no distinct behaviour for Early Stopping on this dataset findable (see fig. 15). Hence, the connection between the paths in the information plane and Early Stopping does not seem to be a general one.
What generalizes perfectly though is that the smaller the bin size used to estimate mutual information, the more the layers tend to be close to the maximum point in the top right. Hence the next section is going to investigate the reasons for this. Also figures 13 and 14 show that the bins empty out and redistribute themselves in the same way as for the Schwartz-Ziv & Tishby (2017) dataset.

[16] Downsampled to 3 classes (1,3 and 8) with 1000 samples each.



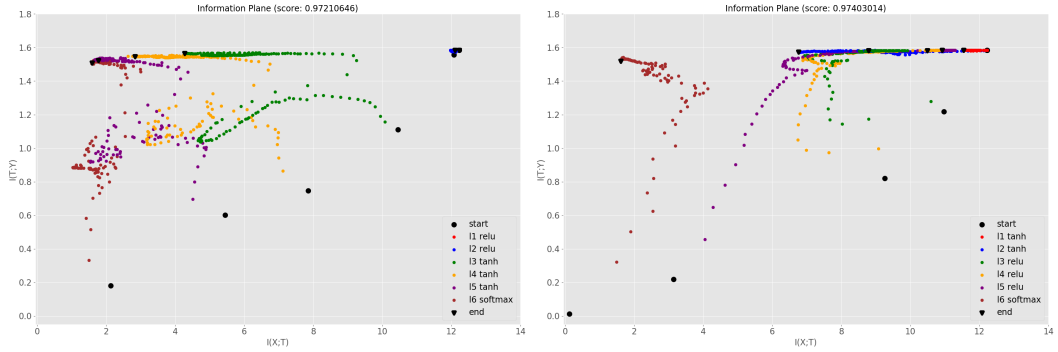
(a) Bin Size 0.01

(b) Bin Size 0.07

(c) Bin Size 0.15

(d) Bin Size 0.3

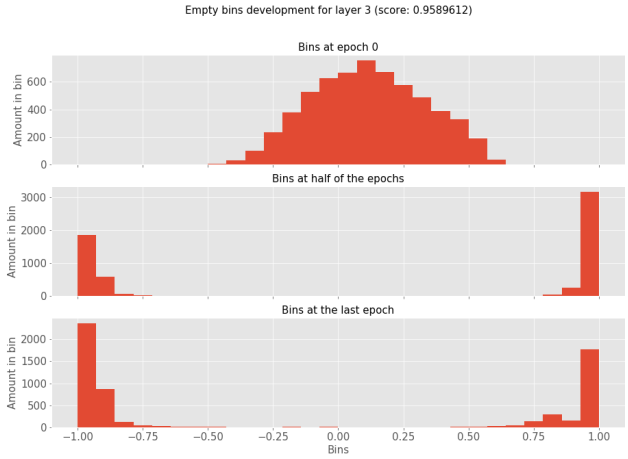
Figure 11: Information planes for different bin sizes for with batch size 256 (MNIST-dataset)



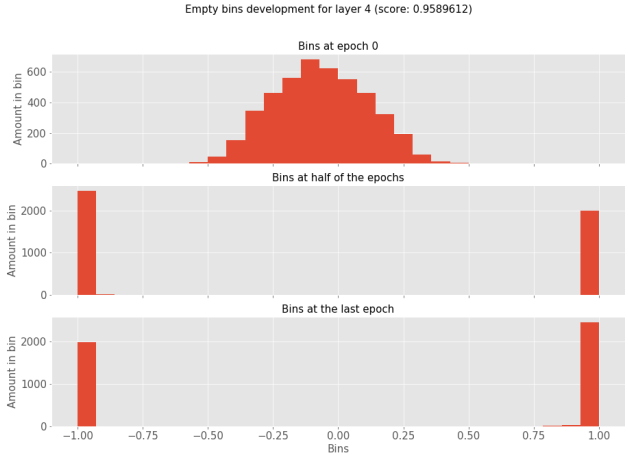
(a) Leading ReLU

(b) Leading TanH

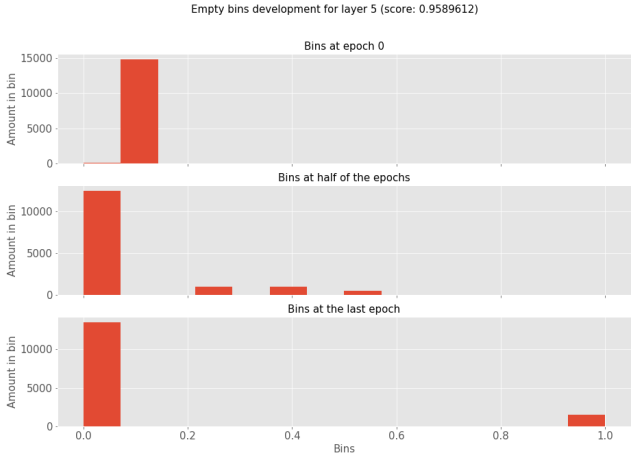
Figure 12: Information plane for Mix-Network for batch size 256 and bin size 0.07 (MNIST dataset)



(a) Layer 4 (TanH)

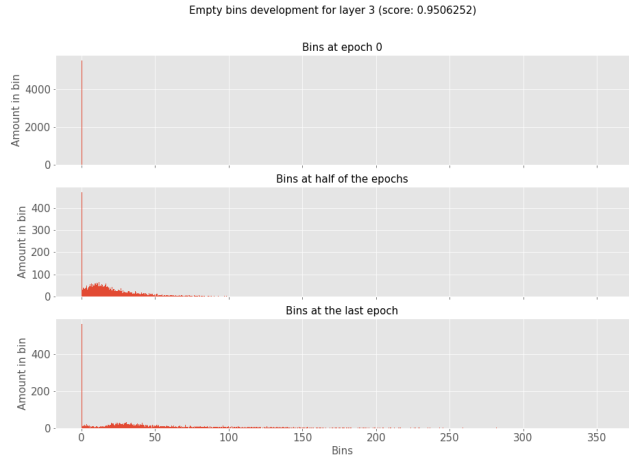


(b) Layer 5 (TanH)

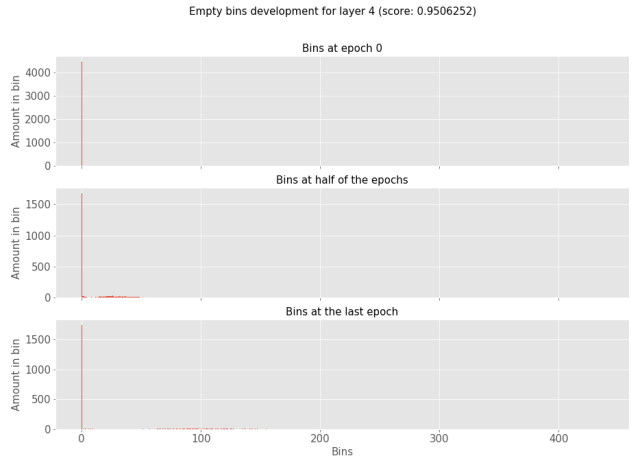


(c) Output Layer (TanH)

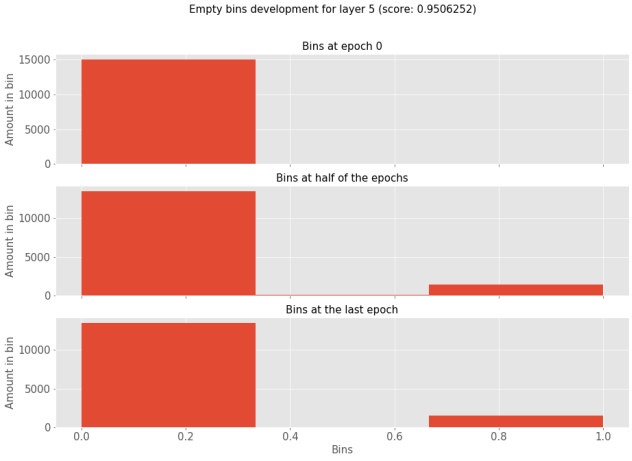
Figure 13: Bin Histograms for TanH-Network at epochs 1, 4000 and 8000 for MNIST-dataset, batch size 256 and bin size 0.07



(a) Layer 4 (ReLU)

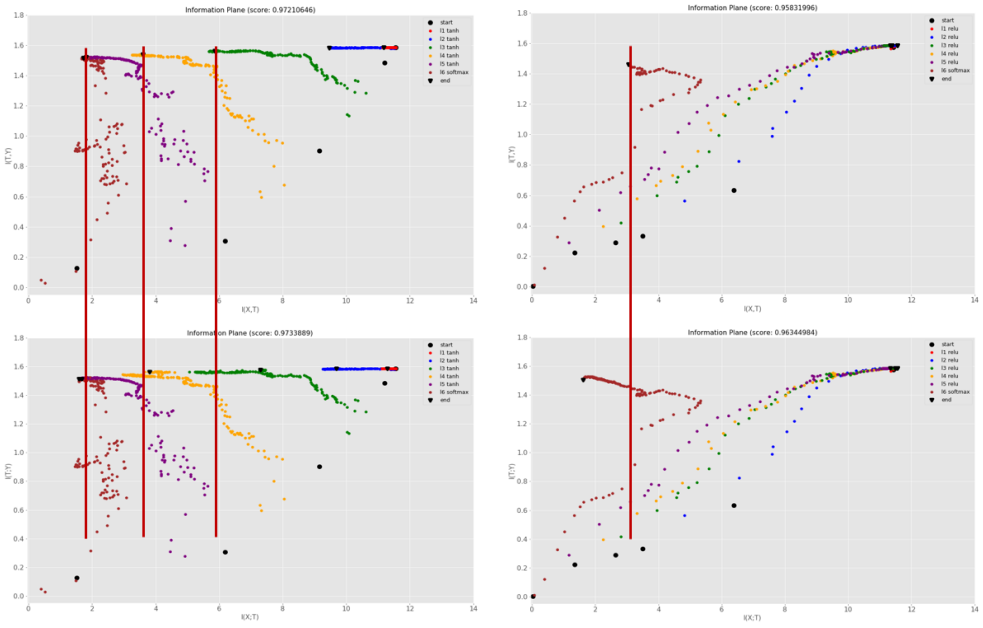


(b) Layer 5 (ReLU)



(c) Output Layer (ReLU)

Figure 14: Bin Histograms for ReLU-Network at epochs 1, 4000 and 8000 for MNIST-dataset, batch size 256 and bin size 0.2 (higher batch size was needed because smaller would lead to too thin bins that cannot be graphed.



(a) TanH (b) ReLU

Figure 15: Comparison of information planes for Early Stopping (top) and Full iterations (bottom) of TanH and ReLU-networks on MNIST-dataset with batch size 256 and bin size 0.07

E IB BOUND

Fig. 16 shows the Information Bound in the information plane and two explanatory achievable compression levels for given $I(\widetilde{X};Y)$.

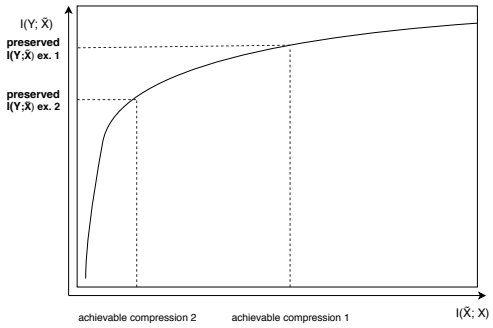
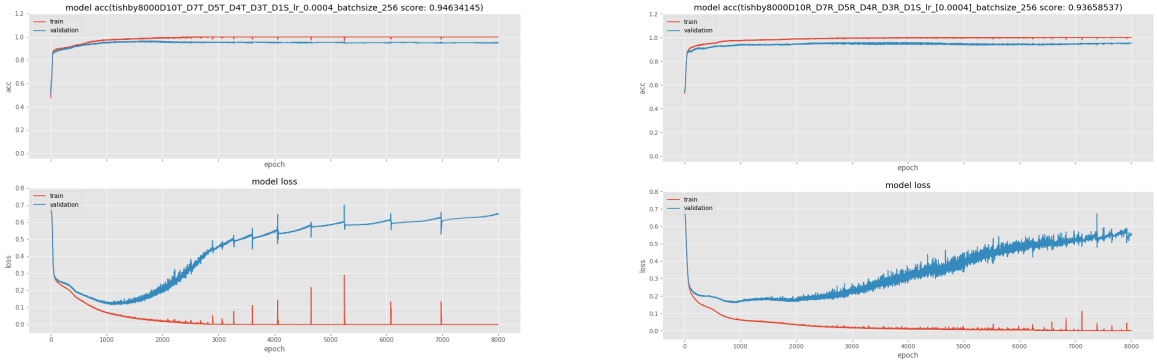


Figure 16: Explanatory Information Plane showing the achievable compression of the message X given a level of mutual information of the message $\widetilde{X}$ and Y. The two dotted lines show two examples of achievable compression levels (representation following Tishby et al. (1999))

F GENERALIZATION CAPABILITIES

In their analysis Schwartz-Ziv & Tishby (2017) focused solely on accuracy as the decisive parameter to judge the generalization capabilities. They argued, that their model was having the best generalization at the point of highest compression. If one takes the cross entropy loss into account as well, the picture is very different. Since cross entropy is an information theoretic metric it makes sense to include it into the analysis if one measures the information flow of a neural network. Figures 17a and 17b show accuracy plotted against cross-entropy loss. One can see that the accuracy of the two networks is indeed very good (scores of 0.946 and 0.937), but if one takes cross entropy into account

it is obvious, that these models are very overfitted. What is shown in the plots, is on the one hand, the model already reached a very good accuracy after roughly 1000 iterations and then just continuous to be very high. On the other hand, the cross entropy loss starts to increase around the same region. This means that the model is actually loosing its confidence. It still assigns almost all the samples correctly as before, but is much more likely to change its mind and assign a sample to the wrong class. Therefore, the model worse at 8000 iterations than it was at 1000. Hence, the maximum compression, which is usually reached at the maximum epoch, cannot be equalled to good generalization.



(a) TanH-Network

(b) ReLU-Network

Figure 17: Training and validation accuracy and cross-entropy loss for networks on Schwartz-Ziv & Tishby (2017)-dataset with batch size 256

G BINNING ESTIMATOR

Schwartz-Ziv & Tishby (2017) use a binning process to estimate the probability distribution of the activations of the different layers with TanH. The idea is to separate the continuous activation range into discrete bins. An example is shown in Figure 18 where a double linearly saturating function is split in 12 bin compartments. For activation functions with known bounds like TanH, these bounds are used to define the "span-width" or the outer borders of the domain. Functions without defined bounds approximate these through usage of the maximum and minimum occurrence in the samples. For each epoch and layer, the recorded activations are then allocated to the bins. And the number of samples in each bin are typically used to estimate the probability distribution. This is a very common approach as it is easy to use (Kraskov et al. (2004)).
Schwartz-Ziv & Tishby (2017) use one sample of the inputs and all of its feature values as a whole, meaning they do not look at the probability of a feature having a certain manifestation, but at the probability of the sample being this feature combination as a whole. To do so the feature manifestations of each sample get "lined-up" as a binary array and the arrays probability is the one actually used. The same happens with the output array and the now discrete (binned) values of the activations.
With these the respective probabilities $P(X)$, $P(Y)$ and $P(T)$ can be calculated. The conditional probabilities $P(T_i, X)$ and $P(T_i, Y)$ are calculated using the inverse of the X and Y arrays and their respective already calculated probabilities. The process is as follows:

1. take the activation data where the inverse of X equals the manifestation of X (meaning where the inverse equals the index of the i for all indices of the probability vector $P(X)$)

2. convert these activations into continuous binary arrays like before

3. calculate the probability of these arrays as $P(T, X)$

With the now estimated probabilities, one can calculate the entropies using the formulas displayed in equations 1 and 2 and with these one can calculate mutual information using equation 4.

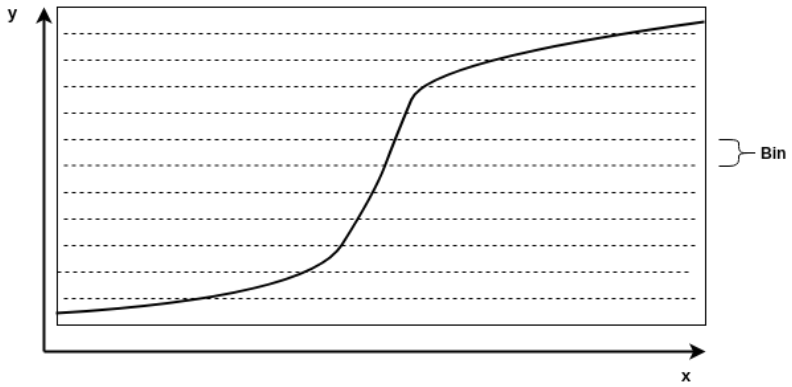


Figure 18: Binning explanation - double linearly saturated function segmented into 12 bins (borders as dotted lines) where x is the sum of the inputs into a neuron and y is the result of the activation

Decreasing the bin size and therefore effectively increasing the number of bins leads to a better approximation of the mutual information as the estimation error converges to 0 (see fig. 19) (Cover & Thomas (2005)—Kraskov et al. (2004)).

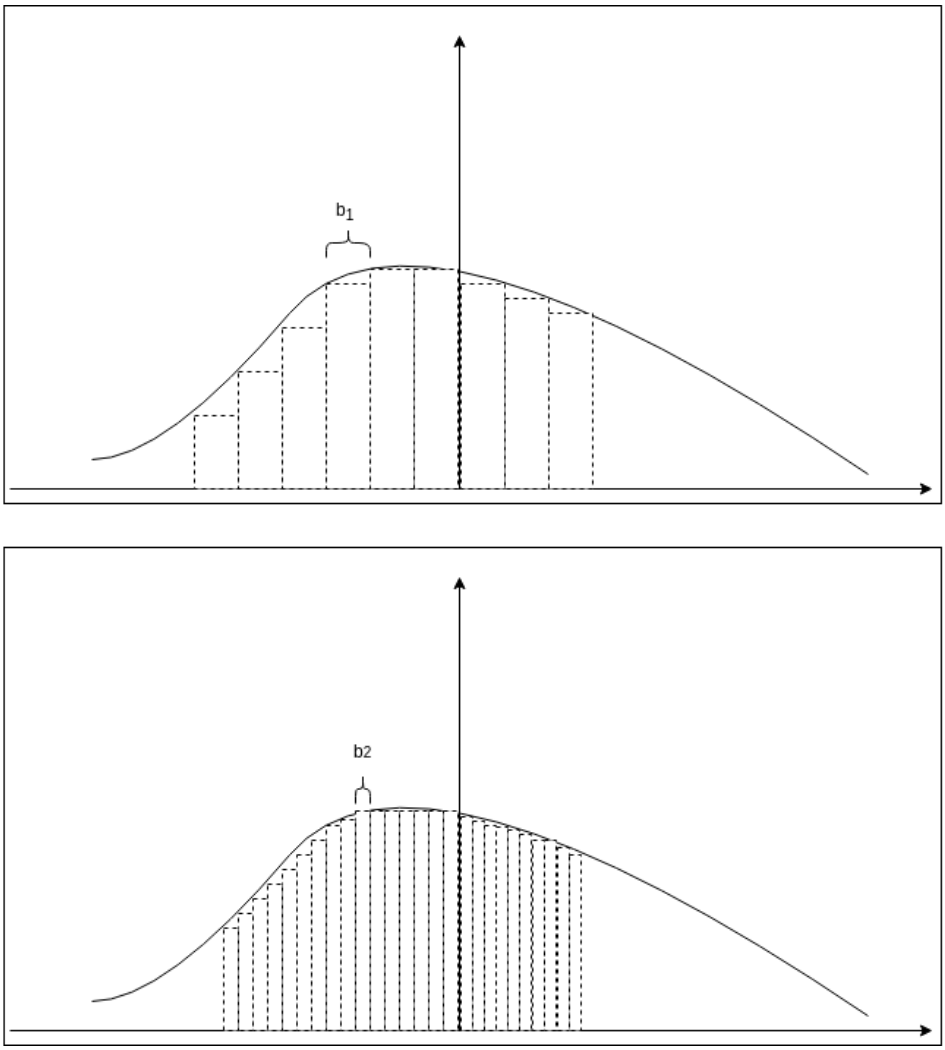


Figure 19: Effect of different bin sizes on estimation error. Here the smaller bin size b_2 allows to approximate the distribution better than b_1 (own representation)