On the Information Bottleneck Theory of Deep Learning

Anonymous authors
Paper under double-blind review

Abstract

The practical successes of deep neural networks have not been matched by theoretical progress that satisfyingly explains their behavior. In this work, we study the information bottleneck (IB) theory of deep learning, which makes three specific claims: first, that deep networks undergo two distinct phases consisting of an initial fitting phase and a subsequent compression phase; second, that the compression phase is causally related to the excellent generalization performance of deep networks; and third, that the compression phase occurs due to the diffusion-like behavior of stochastic gradient descent. Here we show that none of these claims hold true in the general case. Through a combination of analytical results and simulation, we demonstrate that the information plane trajectory is predominantly a function of the neural nonlinearity employed: double-sided saturating nonlinearities like tanh yield a compression phase as neural activations enter the saturation regime, but linear activation functions and single-sided saturating nonlinearities like the widely used ReLU in fact do not. Using recent results on generalization error dynamics in linear neural networks, we show that there is no evident causal connection between compression and generalization: networks that do not compress are still capable of generalization, and vice versa. We also show that the compression phase, when it exists, does not require any stochasticity in training by demonstrating that we can replicate the IB findings using full batch gradient descent rather than stochastic gradient descent. Finally, we show that when an input domain consists of a subset of task-relevant and task-irrelevant information, hidden representations do compress the task-irrelevant information, though the overall information about the input increases, and this compression happens at the same time as the fitting process rather than during a subsequent compression period.

1 Introduction

Deep neural networks (Schmidhuber, 2015; LeCun et al., 2015) are the tool of choice for real-world tasks ranging from visual object recognition (Krizhevsky et al., 2012), to unsupervised learning (Lotter et al., 2016) and reinforcement learning (Silver et al., 2016). These practical successes have spawned many attempts to explain the performance of deep learning systems (Kadmon & Sompolinsky (2016)), mostly in terms of the properties and dynamics of the optimization problem in the space of weights (Saxe et al., 2014; Choromanska et al., 2015; Advani & Saxe, 2017), or the classes of functions that can be efficiently represented by deep networks (Montufar et al., 2014; Poggio et al., 2017). This paper analyzes a recent inventive proposal to study the dynamics of learning through the lens of information theory (Tishby & Zaslavsky, 2015; Shwartz-Ziv & Tishby, 2017). In this view, deep learning is a question of representation learning: each layer of a deep neural network can be seen as a set of summary statistics which contain some but not all of the information present in the input; and retain some but not all of the information about the target output. The amount of information in a hidden layer regarding the input and output can then be measured over the course of learning, yielding a picture of the optimization process in the information plane. Crucially, this method holds the promise to serve as a general analysis that can be used to compare different architectures, using the common currency of mutual information. Moreover, the elegant information bottleneck (IB) theory provides a fundamental bound on the amount of input compression and target output information that any representation can achieve Tishby et al. (2000). The IB bound thus serves as a method-agnostic ideal to which different architectures and algorithms may be compared.

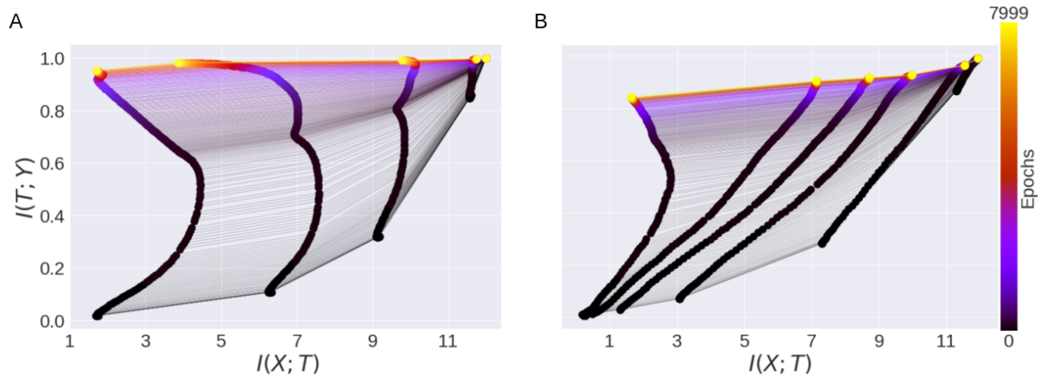

Figure 1: Information plane dynamics and neural nonlinearities. (A) Replication of Shwartz-Ziv & Tishby (2017) for a network with tanh nonlinearities (except for the final layer which contains sigmoidal neurons). The x-axis plots information between each layer and the input, while the y-axis plots information between each layer and the output. The color scale indicates training time in epochs. Each of the six layers produces a curve in the information plane with the input layer at far right, output layer at the far left. Different layers at the same epoch are connected by fine lines. (B) Information plane dynamics with ReLU nonlinearities (except for the final layer of 2 sigmoidal neurons). Here no compression phase is visible in the ReLU layers. For learning curves of both networks, see Appendix A

A preliminary empirical exploration of these ideas in deep neural networks has yielded striking findings (Shwartz-Ziv & Tishby, 2017). Most saliently, trajectories in the information plane appear to consist of two distinct phases: an initial "fitting" phase where mutual information of the hidden layers with both the input and output increases, and a subsequent "compression" phase where mutual information with the input *decreases*. It has been hypothesized that this compression phase is responsible for the excellent generalization performance of deep networks, and further, that this compression phase occurs due to the random diffusion-like behavior of stochastic gradient descent.

Here we study these phenomena using a combination of analytical methods and simulation. In Section 2, we show that the compression observed by Shwartz-Ziv & Tishby (2017) arises primarily due to the double-saturating tanh activation function used. Using simple models, we elucidate the effect of neural nonlinearity on the compression phase. Importantly, we demonstrate that the ReLU activation function, often the nonlinearity of choice in practice, does not exhibit a compression phase. We discuss how this compression via nonlinearity is related to the assumption of noise in the hidden layer representation. To better understand the dynamics of learning in the information plane, in Section 3 we study deep linear networks in a tractable setting where the mutual information can be calculated exactly. We find that deep linear networks do not compress over the course of training for the setting we examine. In Section 4, we investigate the impact of stochasticity in the training process on the information plane dynamics. We train networks with full batch gradient descent, and compare the results to those obtained with stochastic gradient descent. We find comparable compression in both cases, indicating that the stochasticity of SGD is not a primary factor in the observed compression phase. These results may seem difficult to reconcile with the intuition that compression can be necessary to attain good performance: if some input channels primarily convey noise, good generalization requires excluding them. Therefore, in Section 5 we study a situation with explicitly task-relevant and task-irrelevant input dimensions. We show that the hidden-layer mutual information with the task-irrelevant subspace does indeed drop during training, though the overall information with the input increases. However, instead of a secondary compression phase, this task-irrelevant information is compressed at the same time that the task-relevant information is boosted. Our results highlight the importance of noise assumptions in applying information theoretic analyses to deep learning systems, and complicate the IB theory of deep learning by demonstrating instances where representation compression and generalization performance can diverge.

2 Compression and Neural Nonlinearities

The starting point for our analysis is the observation that changing the activation function can markedly change the trajectory of a network in the information plane. In Figure 1A, we show our replication of the result reported by Shwartz-Ziv & Tishby (2017) for networks with the $\tanh$ nonlinearity. This replication was performed with the code supplied by the authors of Shwartz-Ziv & Tishby (2017), and closely follows the experimental setup described therein. Briefly, a neural network with 7 fully connected hidden layers of width 12-10-7-5-4-3-2 is trained with stochastic gradient descent to produce a binary classification from a 12-dimensional input. In our replication we used 256 randomly selected samples per batch. The mutual information of the network layers with respect to the input and output variables is calculated by binning the neuron's tanh output activations into 30 equal intervals between -1 and 1. Discretized values for each neuron in each layer are then used to directly calculate the joint distributions, over the 4096 equally likely input patterns and true output labels. In line with prior work (Shwartz-Ziv & Tishby, 2017), the dynamics in Fig. 1 show a transition between an initial fitting phase, during which information about the input increases, and a subsequent compression phase, during which information about the input decreases.

We then modified the code to train deep networks using rectified linear activation functions ($f(x) = \max(0, x)$). While the activities of $\tanh$ networks are bounded in the range $[-1, 1]$, ReLU networks have potentially unbounded positive activities. To calculate mutual information, we first trained the ReLU networks, next identified their largest activity value over the course of training, and finally chose 100 evenly spaced bins between the minimum and maximum activity values to discretize the hidden layer activity. The resulting information plane dynamics are shown in Fig. 1B. The mutual information with the input monotonically increases in all ReLU layers, with no apparent compression phase. Thus, the choice of nonlinearity substantively affects the dynamics in the information plane.

To understand the impact of neural nonlinearity on the mutual information dynamics, we develop a minimal model that exhibits this phenomenon. In particular, consider the simple three neuron network shown in Fig. 2A. We assume a scalar Gaussian input distribution $X \sim \mathcal{N}(0, 1)$, which is fed through the scalar first layer weight w_1, and passed through a neural nonlinearity $f(\cdot)$, yielding the hidden unit activity $h = f(w_1 X)$. To calculate the mutual information with the input, this hidden unit activity is then binned yielding the new discrete variable $T = \text{bin}(h)$ (for instance, into 30 evenly spaced bins from -1 to 1 for the $\tanh$ nonlinearity). This binning process is depicted in Fig. 2B. In this simple setting, the mutual information $I(T; X)$ between the binned hidden layer activity T and the input X can be calculated exactly. In particular,

$$\begin{aligned} I(T;X) &= H(T) - H(T|X) && (1)\\ &= H(T) && (2)\\ &= -\sum_{i=1}^{N} p_i \log p_i && (3) \end{aligned}$$

where $H(\cdot)$ denotes entropy, and we have used the fact that $H(T|X) = 0$ since T is a deterministic function of X. Here the probabilities $p_i = P(h \geq b_i \text{ and } h < b_{i+1})$ are simply the probability that an input X produces a hidden unit activity that lands in bin i, defined by lower and upper bin limits b_i and b_{i+1} respectively. This probability can be calculated exactly for monotonic nonlinearities $f(\cdot)$ using the cumulative density of X,

$$p_i = P(X \geq f^{-1}(b_i)/w_1 \text{ and } X < f^{-1}(b_{i+1})/w_1), \tag{4}$$

where $f^{-1}(\cdot)$ is the inverse function of $f(\cdot)$.

As shown in Fig. 2C-D, as a function of the weight w_1, mutual information with the input first increases and then decreases for the $\tanh$ nonlinearity, but always increases for the ReLU nonlinearity. Intuitively, for small weights $w_1 \approx 0$, neural activities lie near zero on the approximately linear part of the $\tanh$ function. Therefore $f(w_1 X) \approx w_1 X$, yielding a rescaled Gaussian with information that grows with the size of the weights. However for very large weights $w_1 \to \infty$, the $\tanh$ hidden unit nearly always saturates, yielding a discrete variable that concentrates in just two bins. This is more or less a coin flip, containing mutual information with the input of approximately 1 bit. Hence the distribution of T collapses to a much lower entropy distribution, yielding compression for large weight values. With the ReLU nonlinearity, half of the inputs are negative and land in the bin

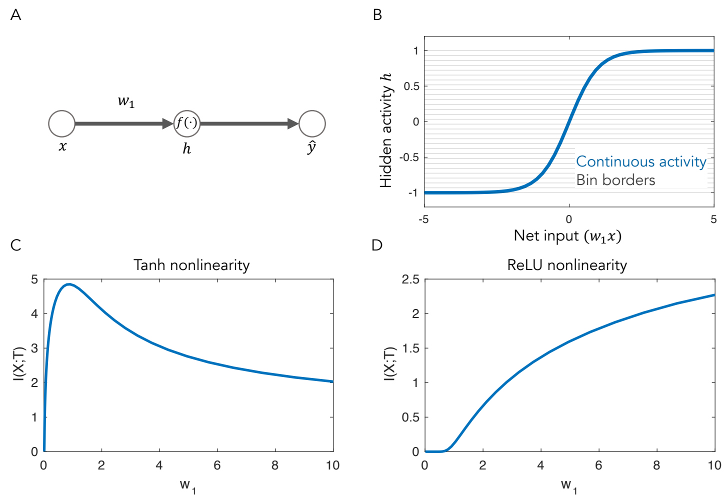

Figure 2: Nonlinear compression in a minimal model. (A) A three neuron nonlinear network which receives Gaussian inputs x, multiplies by weight w_1, and maps through neural nonlinearity $f(\cdot)$ to produce hidden unit activity h. (B) The continuous activity h is binned into a discrete variable T for the purpose of calculating mutual information. Blue: continuous $\tanh$ nonlinear activation function. Grey: Bin borders for 30 bins evenly spaced between -1 and 1. Because of the saturation in the sigmoid, a wide range of large magnitude net input values map to the same bin. (C) Mutual information with the input as a function of weight size w_1 for a $\tanh$ nonlinearity. Information increases for small w_1 and then decreases for large w_1 as all inputs land in one of the two bins corresponding to the saturation regions. (D) Mutual information with the input for the ReLU nonlinearity increases without bound. Half of all inputs land in the bin corresponding to zero activity, while the other half have information that scales with the size of the weights.

containing a hidden activity of zero. The other half are Gaussian distributed, and thus have entropy that increases with the size of the weight.

Hence double-saturating nonlinearities can lead to compression of information about the input, as hidden units enter their saturation regime, due to the binning procedure used to calculate mutual information. We note that this binning procedure can be viewed as implicitly adding noise to the hidden layer activity: a range of X values map to a single bin, such that the mapping between X and T is no longer perfectly invertible (Laughlin, 1981). Without binning, there would be no loss of information about the input provided that the nonlinearity is invertible, and the mutual information $I(h; X)$ between the continuous hidden activity h and X would be infinite. Hence the binning procedure is crucial to the information theoretic analysis, and corresponds approximately to a model where noise enters the system after the calculation of h, that is, $T = h + \epsilon$, where ϵ is noise of fixed variance independent from h and x. This approach is common in information theoretic analyses of deterministic systems, and can serve as a measure of the complexity of a system's representation (see Sec 2.4 of Shwartz-Ziv & Tishby (2017)). However, this noise is not added in practice either during training or testing in these neural networks. It therefore remains unclear whether robustness of a representation to this sort of noise in fact influences generalization performance in deep learning systems. Furthermore, the addition of noise means that different architectures may no longer be compared in a common currency of mutual information: architectures that implement an identical input-output map can nevertheless have different robustness to noise added in their internal representation. For instance, in the linear case, first layer weights could be small and second layer weights could be large, making a model highly sensitive to noise in the hidden layer; or visa versa. Yet these networks could compute exactly the same input-output map and therefore generalize identically. Finally, we note that approaches which view the weights obtained from the training process as the random variables of interest may sidestep this issue (Achille & Soatto, 2017).

Hence when a $\tanh$ network is initialized with small weights and over the course of training comes to saturate its nonlinear units (as it must to compute most functions of practical interest), it will

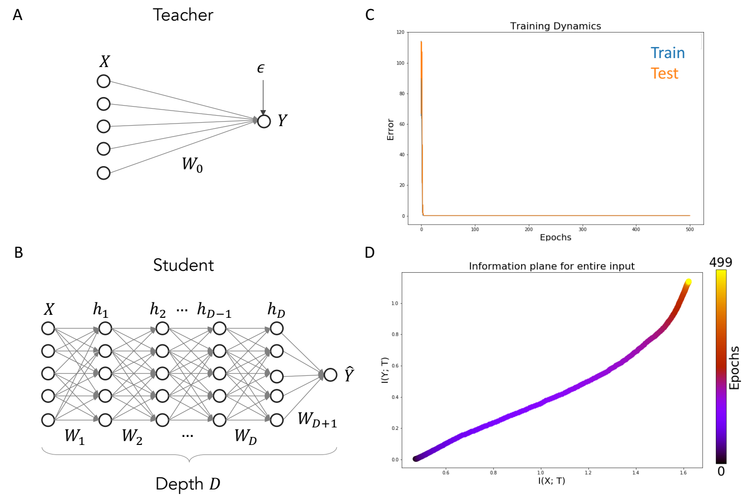

Figure 3: Generalization and information plane dynamics in deep linear networks. (A) A linear teacher network generates a dataset by passing Gaussian inputs X through its weights and adding noise. (B) A deep linear student network is trained on the dataset (here the network has 1 hidden layer to allow comparison with Fig. 4A, see Supplementary Figure 10 for a deeper network). (C) Training and testing error over time. (D) Information plane dynamics. No compression is observed.

enter a compression period where mutual information decreases. Figures 8-9 of Appendix B show histograms of neural activity over the course of training, demonstrating that activities in the $\tanh$ network enter the saturation regime over training. This nonlinearity-based compression furnishes another explanation for the observation that training slows down as $\tanh$ networks enter their compression phase (Shwartz-Ziv & Tishby, 2017): some fraction of inputs have saturated the nonlinearities, reducing backpropagated error gradients.

3 Information Plane Dynamics in Deep Linear Networks

The preceding section investigates the role of nonlinearity in the observed compression behavior, tracing the source to double-saturating nonlinearities and the binning methodology used to calculate mutual information. However, other mechanisms could lead to compression as well. Even without nonlinearity, neurons could converge to highly correlated weights, or project out irrelevant directions of the input. These phenomena are not possible to observe in our simple three neuron minimal model, as they require multiple inputs and hidden layer activities. To search for these mechanisms, we turn to a tractable model system: deep linear neural networks (Baldi & Hornik (1989); Fukumizu (1998); Saxe et al. (2014)). In particular, we exploit recent results on the generalization dynamics in simple linear networks trained in a student-teacher setup (Seung et al., 1992; Advani & Saxe, 2017). This setting allows exact calculation of the generalization performance of the network, exact calculation of the mutual information of the representation (without any binning procedure), and, though we do not do so here, direct comparison to the IB bound which is already known for linear Gaussian problems (Chechik et al., 2005).

We consider a scenario where a linear teacher neural network generates input and output examples which are then fed to a deep linear student network to learn (Fig. 3A). Following the formulation of (Advani & Saxe, 2017), we assume multivariate Gaussian inputs $X \sim \mathcal{N}(0, \frac{1}{N_i} I_{N_i})$ and a scalar output Y. The output is generated by the teacher network according to

$$Y = W_o X + \epsilon_o \tag{5}$$

where $\epsilon_o \sim \mathcal{N}(0, \sigma_o^2)$ is output noise and the teacher weights W_o are drawn independently from $\mathcal{N}(0, \sigma_w^2)$. Here, the weights of the teacher define the rule to be learned. The signal to noise ratio $\mathrm{SNR} = \sigma_w^2/\sigma_o^2$ determines the strength of the rule linking inputs to outputs.

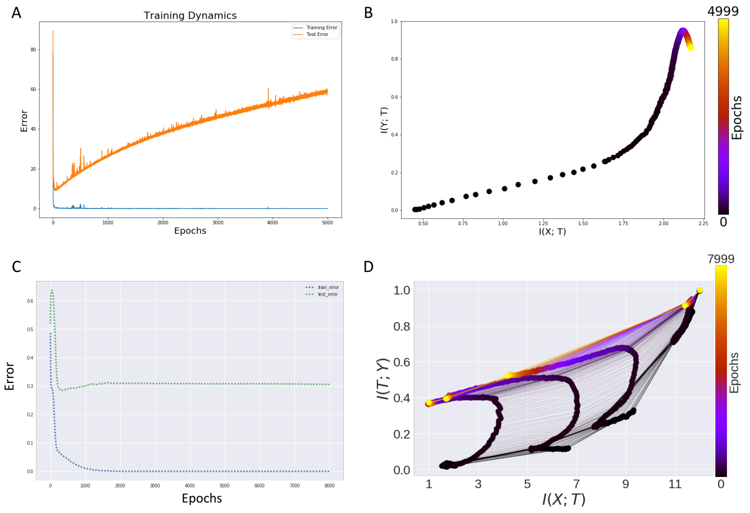

Figure 4: Overtraining and information plane dynamics. (A) Average training and test mean square error for a deep linear network trained with SGD. Overtraining is substantial. Other parameters: N_i = 100, P = 100, Number of hidden units = 100, Batch size = 5 (B) Information plane dynamics. No compression is observed, and information about the labels is lost during overtraining. (C) Average train and test accuracy (% correct) for nonlinear $\tanh$ networks exhibiting modest overfitting ($N = 8$). (D) Information plane dynamics. Overfitting occurs despite continued compression.

To train the student network, a dataset of P examples is generated using the teacher. The student network is then trained to minimize the mean squared error between its output and the target output using standard (batch or stochastic) gradient descent on this dataset. Here the student is a deep linear neural network consisting of potentially many layers, but where the the activation function of each neuron is simply $f(u) = u$. That is, a depth D deep linear network computes the output $\hat{Y} = W_{D+1}W_D \cdots W_2 W_1 X$. While linear activation functions stop the network from computing complex nonlinear functions of the input, deep linear networks nevertheless show complicated nonlinear learning trajectories (Saxe et al., 2014), the optimization problem remains nonconvex (Baldi & Hornik, 1989), and the generalization dynamics can exhibit substantial overtraining (Fukumizu, 1998; Advani & Saxe, 2017).

Importantly, because of the simplified setting considered here, the true generalization error is easily shown to be

$$E_g(t) = ||W_o - W_{tot}(t)||_F^2 + \sigma_o^2 \tag{6}$$

where $W_{tot}(t)$ is the overall linear map implemented by the network at training epoch t (that is, $W_{tot} = W_{D+1}W_D \cdots W_2 W_1$).

Furthermore, the mutual information with the input and output may be calculated exactly, because the distribution of the activity of any hidden layer is Gaussian. Let T be the activity of a specific hidden layer, and let $\bar{W}$ be the linear map from the input to this activity (that is, for layer l, $\bar{W} = W_l \cdots W_2 W_1$). Since $T = \bar{W}X$, the mutual information of X and T calculated using differential entropy is infinite. For the purpose of calculating the mutual information, therefore, we assume that Gaussian noise is added to the hidden layer activity, $T = \bar{W}X + \epsilon_{MI}$, with mean 0 and variance $\sigma_{MI}^2 = 1.0$. This allows the analysis to apply to networks of any size, including overcomplete layers, but as before we emphasize that we do not add this noise either during training or testing. With these assumptions, T and X are jointly Gaussian and we have

$$I(T; X) = \log|\bar{W}\bar{W}^T + \sigma_{MI}^2 I_{N_h}| - \log|\sigma_{MI}^2 I_{N_h}| \tag{7}$$

where $|\cdot|$ denotes the determinant of a matrix. Finally the mutual information with the output Y, also jointly Gaussian, can be calculated similarly (see Eqns. (8)-(11) of Appendix D).

Fig. 3 shows example training and test dynamics over the course of learning in panel C, and the information plane dynamics in panel D. Here the network has an input layer of 100 units, 1 hidden

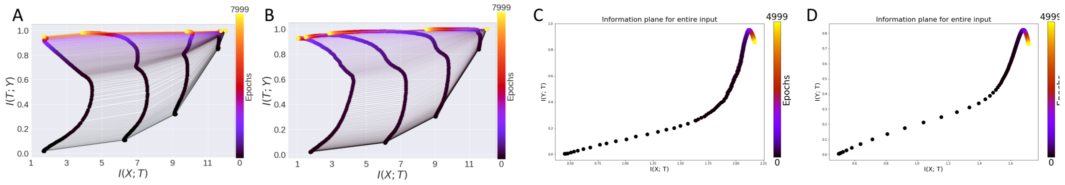

Figure 5: Stochastic training and the information plane. (A) tanh network trained with SGD. (B) tanh network trained with BGD. (C) Linear network trained with SGD. (D) Linear network trained with BGD. Both random and non-random training procedures show similar information plane dynamics.

layer of 100 units each and one output unit. The network was trained with batch gradient descent on a dataset of 100 examples drawn from the teacher with signal to noise ratio of 1.0. The linear network behaves qualitatively like the ReLU network, and does not exhibit compression. Nevertheless, it learns a map that generalizes well on this task and shows minimal overtraining. Hence, in the setting we study here, generalization performance can be acceptable without any compression phase.

The results in (Advani & Saxe (2017)) show that, for the case of linear networks, overtraining is worst when the number of inputs matches the number of training samples, and is reduced by making the number of samples smaller or larger. Fig. 4 shows learning dynamics with the number of samples matched to the size of the network. Here overfitting is substantial, and again no compression is seen in the information plane. Comparing to the result in Fig. 3D, both networks exhibit similar information dynamics with respect to the input (no compression), but yield different generalization performance.

Hence, in this linear analysis of a generic setting, there do not appear to be additional mechanisms that cause compression over the course of learning; and generalization behavior can be widely different for networks with the same dynamics of information compression regarding the input. We note that, in the setting considered here, all input dimensions have the same variance, and the weights of the teacher are drawn independently. Because of this, there are no special directions in the input, and each subspace of the input contains as much information as any other. It is possible that, in real world tasks, higher variance inputs are also the most likely to be relevant to the task (here, have large weights in the teacher). We have not investigated this possibility here.

To see whether similar behavior arises in nonlinear networks, we trained tanh networks in the same setting as Section 2, but with 30% of the data, which we found to lead to modest overtraining. Fig. 4C-D shows the resulting train, test, and information plane dynamics. Here the tanh networks show substantial compression, despite exhibiting overtraining. This establishes a dissociation between behavior in the information plane and generalization dynamics: networks that compress may (Fig. 1A) or may not (Fig. 4C-D) generalize well, and networks that do not compress may (Figs.1B, 3A-B) or may not (Fig. 4A-B) generalize well.

4 Compression in Batch Gradient Descent and SGD

Next, we ask whether the stochastic nature of the most popular training algorithm for deep networks, stochastic gradient descent, contributes to compression. Because the choice of example in stochastic gradient descent is random, the weight updates over time contain an element of randomness that may behave like a diffusion in weight space (Shwartz-Ziv & Tishby, 2017).

To interrogate the role of randomness in compression, we consider two distinct training procedures: offline stochastic gradient descent (SGD), which learns from a fixed-size dataset, and updates weights by repeatedly sampling a single example from the dataset and calculating the gradient of the error with respect to that single sample (the typical procedure used in practice); and batch gradient descent (BGD), which learns from a fixed-size dataset, and updates weights using the gradient of the total error across all examples. Batch gradient descent uses the full training dataset and, crucially, therefore has no randomness or diffusion-like behavior in its updates.

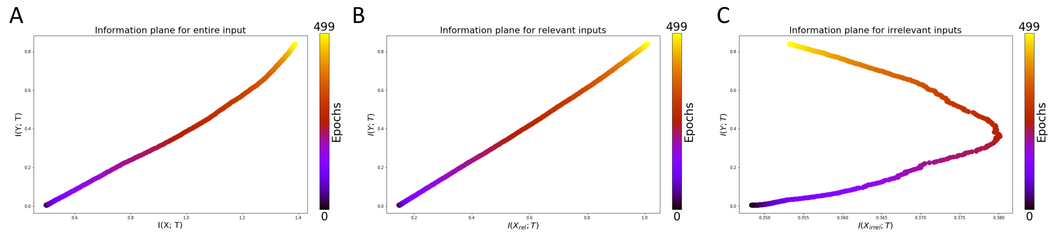

Figure 6: Simultaneous fitting and compression. (A) For a task with a large task-irrelevant subspace in the input, a linear network shows no overall compression of information about the input. (B) The information with the task-relevant subspace increases robustly over training. (C) However, the information specifically about the task-irrelevant subspace does compress after initially growing as the network is trained.

We trained $\tanh$ and linear networks with SGD and BGD and compare their information plane dynamics in Fig. 5. We find largely consistent information dynamics in both instances, with robust compression in $\tanh$ networks for both methods. Thus randomness in the training process does not appear to contribute substantially to compression of information about the input. This finding is consistent with the view presented in Section 2 that compression arises predominantly from the double saturating nonlinearity.

5 Simultaneous Fitting and Compression

Our finding that generalization can occur without compression may seem difficult to reconcile with the intuition that certain tasks involve suppressing irrelevant directions of the input. In the extreme, if certain inputs contribute nothing but noise, then good generalization requires ignoring them. To study this, we consider a variant on the linear student-teacher setup of Section 3: we partition the input X into a set of task-relevant inputs X_{rel} and a set of task-irrelevant inputs X_{irrel}, and alter the teacher network so that the teacher's weights to the task-irrelevant inputs are all zero. Hence the inputs X_{irrel} contribute only noise, while the X_{rel} contain signal. We then calculate the information plane dynamics for the whole layer, and for the task-relevant and task-irrelevant inputs separately. Fig. 6 shows information plane dynamics for a deep linear neural network trained using SGD (5 samples/batch) on a task with 30 task-relevant inputs and 70 task-irrelevant inputs. While the overall dynamics show no compression phase, the information specifically about the task-irrelevant subspace does compress over the course of training. This compression process occurs at the same time as the fitting to the task-relevant information. Thus, when a task requires ignoring some inputs, the information with these inputs specifically will indeed be reduced; but overall mutual information with the input in general may still increase.

6 Discussion

Our results suggest that compression dynamics in the information plane are not a general feature of deep networks, but are critically influenced by the nonlinearities employed by the network. Double-saturating nonlinearities cause compression due to the binning procedure, while single-sided saturating nonlinearities like ReLUs do not compress in general. Consistent with this view, we find that stochasticity in the training process does not contribute to compression in the cases we investigate. Furthermore, we have found instances where generalization performance does not clearly track information plane behavior, questioning the causal link between compression and generalization. Hence information compression may parallel the situation with sharp minima: although empirical evidence has shown a correlation with generalization error in certain settings and architectures, further theoretical analysis has shown that sharp minima can in fact generalize well (Dinh et al., 2017). We emphasize that compression still may occur within a subset of the input dimensions if the task demands it. This compression, however, is interleaved rather than in a secondary phase and may not be visible by information metrics that track the overall information between a hidden layer and the input.

REFERENCES

A. Achille and S. Soatto. On the Emergence of Invariance and Disentangling in Deep Representations. *arXiv*, 2017. URL http://arxiv.org/abs/1706.01350.

Madhu S Advani and Andrew M Saxe. High-dimensional dynamics of generalization error in neural networks. *arXiv preprint arXiv:1710.03667*, 2017.

P. Baldi and K. Hornik. Neural networks and principal component analysis: Learning from examples without local minima. *Neural Networks*, 2(1):53–58, jan 1989. ISSN 08936080. doi: 10.1016/0893-6080(89)90014-2. URL http://linkinghub.elsevier.com/retrieve/pii/0893608089900142.

Gal Chechik, Amir Globerson, Naftali Tishby, and Yair Weiss. Information bottleneck for gaussian variables. *J. Mach. Learn. Res.*, 6(Jan):165–188, 2005.

A. Choromanska, M. Henaff, M. Mathieu, G. Ben Arous, and Y. LeCun. The Loss Surfaces of Multilayer Networks. In *Proceedings of the 18th International Conference on Artificial Intelligence*, volume 38, 2015.

L. Dinh, R. Pascanu, S. Bengio, and Y. Bengio. Sharp Minima Can Generalize For Deep Nets. In *ICML*, 2017.

K. Fukumizu. Effect of Batch Learning In Multilayer Neural Networks. In *Proceedings of the 5th International Conference on Neural Information Processing*, pp. 67–70, 1998.

J. Kadmon and H. Sompolinsky. Optimal Architectures in a Solvable Model of Deep Networks. In *NIPS*, 2016. URL https://papers.nips.cc/paper/6330-optimal-architectures-in-a-solvable-model-of-deep-networks.pdf.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet classification with deep convolutional neural networks. In F Pereira, C J C Burges, L Bottou, and K Q Weinberger (eds.), *Advances in Neural Information Processing Systems 25*, pp. 1097–1105. Curran Associates, Inc., 2012.

S. Laughlin. A simple coding procedure enhances a neuron's information capacity, 1981. ISSN 18657125.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521:436–444, 2015. doi: 10.1038/nature14539.

William Lotter, Gabriel Kreiman, and David D. Cox. Deep predictive coding networks for video prediction and unsupervised learning. *CoRR*, abs/1605.08104, 2016. URL http://arxiv.org/abs/1605.08104.

G. Montufar, R. Pascanu, K. Cho, and Y. Bengio. On the Number of Linear Regions of Deep Neural Networks. In *NIPS*, 2014.

H. Poggio, T.and Mhaskar, L. Rosasco, B. Miranda, and Q. Liao. Why and when can deep-but not shallow-networks avoid the curse of dimensionality: A review. *International Journal of Automation and Computing*, pp. 1–17, 2017. ISSN 1476-8186. doi: 10.1007/s11633-017-1054-2. URL http://link.springer.com/10.1007/s11633-017-1054-2.

A.M. Saxe, J.L. McClelland, and S. Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In Y. Bengio and Y. LeCun (eds.), *the International Conference on Learning Representations*, Banff, Canada, 2014.

J. Schmidhuber. Deep Learning in Neural Networks: An Overview. *Neural Networks*, 61:85–117, 2015. ISSN 08936080. doi: 10.1016/j.neunet.2014.09.003. URL http://arxiv.org/abs/1404.7828.

H.S. Seung, H. Sompolinsky, and N. Tishby. Statistical mechanics of learning from examples. *Physical Review A*, 45(8):6056–6091, 1992. ISSN 1050-2947. doi: 10.1103/PhysRevA.45.6056. URL http://www.ncbi.nlm.nih.gov/pubmed/9907706{%}5Cnhttp://link.aps.org/doi/10.1103/PhysRevA.45.6056.

Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *CoRR*, abs/1703.00810, 2017. URL `http://arxiv.org/abs/1703.00810`.

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016. ISSN 0028-0836. doi: 10.1038/nature16961. URL `http://www.nature.com/doifinder/10.1038/nature16961`.

N. Tishby and N. Zaslavsky. Deep learning and the information bottleneck principle. In *IEEE Information Theory Workshop (ITW)*. IEEE, apr 2015. ISBN 978-1-4799-5524-4. doi: 10.1109/ITW.2015.7133169.

N. Tishby, F.C. Pereira, and W. Bialek. The information bottleneck method. *arxiv.org*, 2000.

A LEARNING CURVES FOR $\tanh$ AND RELU NETWORKS

Supplementary Figure 7 shows the learning curves for $\tanh$ and ReLU networks depicted in Fig. 1.

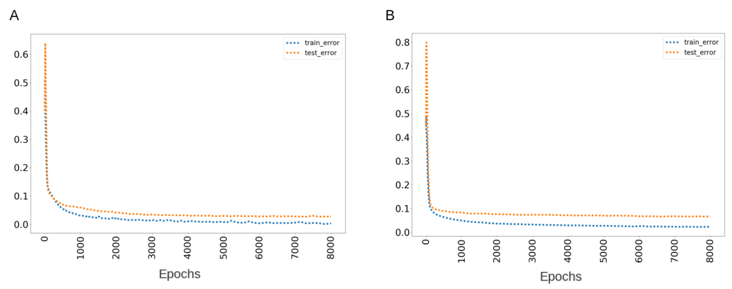

Figure 7: Learning curves for A) $\tanh$ neural network in 1 A and B) ReLU neural network in 1 B. Both networks show good generalization with regards to the test data.

B HISTOGRAMS OF NEURAL ACTIVATIONS

Supplementary Figures 8 and 9 show histograms of neural activities over the course of training in $\tanh$ and ReLU networks respectively.

C INFORMATION PLANE DYNAMICS IN DEEPER LINEAR NETWORKS

Supplementary Figure 10 shows information plane dynamics for a deep neural network with five hidden layers.

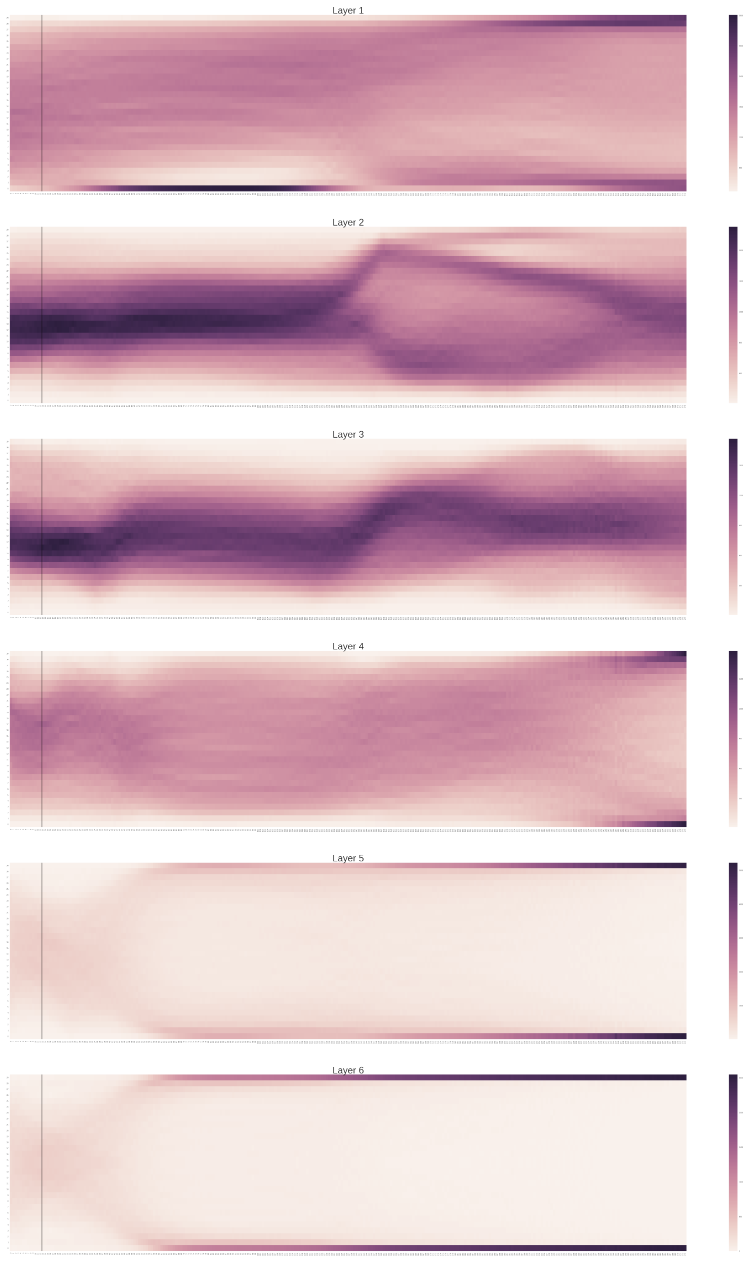

Figure 8: Histogram of neural activities in a $\tanh$ network during training. The final three layers eventually saturate in the top and bottom bins corresponding to the saturation limits of the $\tanh$ activation function, explaining the compression observed in $\tanh$. X-axis: training time in epochs. Y-axis: Hidden activity bin values from lowest to highest. Colormap: density of hidden layer activities across all input examples.

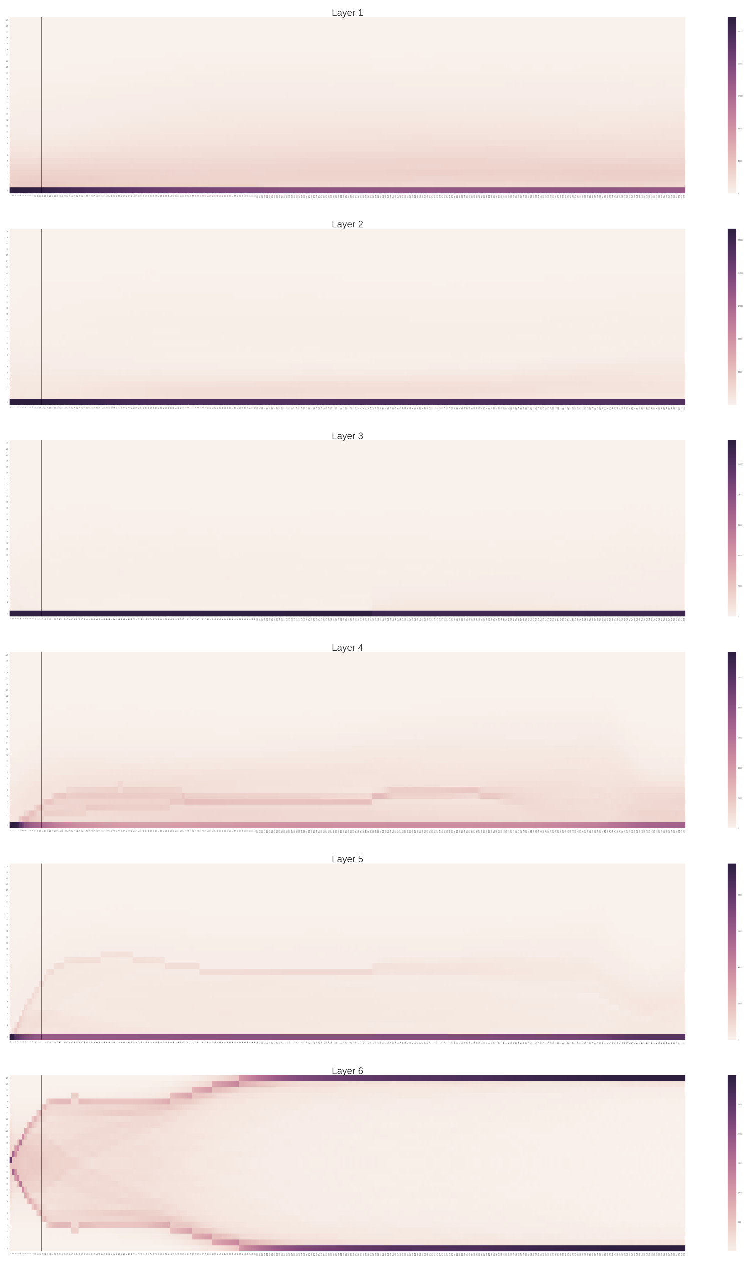

Figure 9: Histogram of neural activities in a ReLU network during training. ReLU layers 1-5 have a roughly constant fraction of activities at zero, corresponding to instances where the ReLU is off; the nonzero activities disperse over the course of training without bound, yielding higher entropy distributions. The sigmoid output layer 6 converges to its saturation limits, and is the only layer that compresses during training (c.f. Fig. 1B). X-axis: training time in epochs. Y-axis: Hidden activity value. Colormap: density of hidden layer activities across all input examples.

D Linear mutual information calculation

For the linear setting considered here, the mutual information between a hidden representation T and the output Y may be calculated using the relations

$$H(Y) = \frac{N_o}{2}\log(2\pi e) + \frac{1}{2}\log|W_o W_o^T + \sigma_o^2 I_{N_o}|, \tag{8}$$

$$H(T) = \frac{N_h}{2}\log(2\pi e) + \frac{1}{2}\log|\bar{W}\bar{W}^T + \sigma_{MI}^2 I_{N_h}|, \tag{9}$$

$$H(Y;T) = \frac{N_o + N_h}{2}\log(2\pi e) + \frac{1}{2}\log\left|\begin{matrix} \bar{W}\bar{W}^T + \sigma_{MI}^2 I_{N_h} & \bar{W}W_o^T, \\ W_o\bar{W}^T & W_o W_o^T + \sigma_o^2 I_{N_h} \end{matrix}\right|, \tag{10}$$

$$I(Y;T) = H(Y) + H(T) - H(Y;T). \tag{11}$$

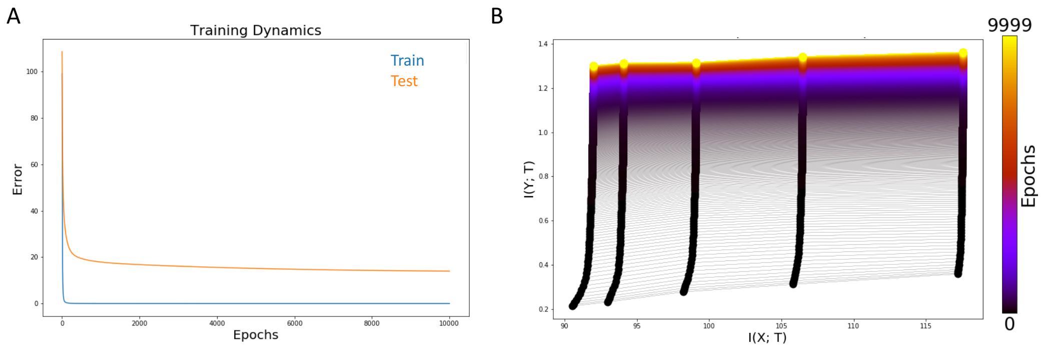

Figure 10: Information plane dynamics in a deep linear neural network. (A) Train and test error during learning. (B) Information plane dynamics. No compression is visible.