Direct Validation of the Information Bottleneck Principle for Deep Nets

Adar Elad[*], Doron Haviv[*], Yochai Blau, Tomer Michaeli
Technion – Israel Institute of Technology, Haifa, Israel

Abstract

The information bottleneck (IB) has been suggested as a fundamental principle governing performance in deep neural nets (DNNs). This idea sparked research on the information plane dynamics during training with the cross-entropy loss, and on using the IB of some "bottleneck" layer as a loss function. However, the claim that reaching the maximal value of the IB Lagrangian in each layer leads to optimal performance, was in fact never directly confirmed. In this paper, we propose a direct way of validating this hypothesis, using layer-by-layer training with the IB loss. In accordance with the original theory, we train each DNN layer explicitly with the IB objective (and without any classification loss), and freeze it before moving on to train the next layer. While mutual information (MI) is generally hard to estimate in high dimensions, we show that in the case of MI between DNN layers, this can be done quite accurately using a modification of the recently proposed mutual information neural estimator [4]. Interestingly, we find that layer-by-layer training with the IB loss leads to accuracy which is on-par with end-to-end training with the cross entropy loss. This is, thus, the first direct experimental illustration of the link between the IB value in each layer, and a net's performance.

1. Introduction

Deep neural nets (DNN) have shown unparalleled success in many fields, yet the theoretical understanding of these models is lagging behind. Among the various attempts to understand their inner-workings, Tishby and Zaslavsky [15] suggested a link to the Information Bottleneck (IB) principle of Tishby et al. [14]. Their theory claims that optimal performance is attained if *each layer* simultaneously attempts to maximize its mutual information (MI) with the target space, while minimizing its MI with the input space. The logic is that the DNN layers should successively compress the latent representation so as to allow for good generalization, while retaining only the information essential for target prediction. Specifically, it was suggested that each DNN layer should maximize the IB Lagrangian

$$\mathcal{L}_{\text{IB}} = I(Y; L_i) - \beta I(X; L_i), \tag{1}$$

where I denotes MI, X and Y are the input and target random variables, L_i is the latent representation at the ith hidden layer, and β is a trade-off parameter.

Various papers studied this principle or used it for regularization in DNN training[1]. In particular, some works studied the dynamics of each of the terms in (1) during training with the popular cross-entropy (CE) loss [13, 12]. While the IB functional seems to steadily increase along training iterations, such experimental results are highly dependent on the method used to estimate the MI [12], and thus do not serve as a direct validation of the original claim of [15]. Several works used the IB objective (1) as a loss function for training, but only on a *single* "bottleneck" layer [2, 4, 8] and not on each layer separately as in the original theory. Also, due to practical difficulties in estimating the MI between DNN layers, most works focus on *stochastic* nets trained in an end-to-end fashion [1, 2, 4, 8]. Therefore, the fundamental claim that optimal performance is achieved when *every* layer approaches the maximal value of the IB objective, has not been explicitly confirmed to date. Our goal in this paper is to experimentally validate this hypothesis, in particular for the more popular *deterministic* DNNs.

It should be emphasized that we are *not* interested in analyzing the information plane dynamics of deep nets trained with the CE loss, as in [13, 12]. We are rather interested in a much more fundamental question, which is independent of how a net is trained: Does a deep architecture achieve optimal performance when the IB value in each of its layers is maximal? Our way

[*]Equal contribution.

[1]An inclusive survey of related work appears in the supplementary material.

to directly validate this hypothesis of Tishby and Zaslavsky [15], is to train a DNN while guaranteeing that *each* layer optimizes the IB functional in (1). Starting from the first hidden layer, we train each layer independently and freeze it before moving on to the next layer. Surprisingly, while layer-by-layer training is obviously sub-optimal in general, we find that when done with the IB functional, the performance is not only comparable but also often better than end-to-end training with the CE loss. This is thus the first explicit confirmation of the merit of the IB objective applied to each layer independently, as hypothesized in [15].

Strictly speaking, direct use of the IB functional for *deterministic* nets with continuous inputs is meaningless, as the term $I(X;L_i)$ in (1) is always infinite [12, 3]. This problem is mitigated if one resorts to *stochastic* DNNs via the introduction of noise after each layer, which ensures that $I(X;L_i)$ is finite [12, 3], as done in [2, 4, 8, 1, 5]. However, we are interested in deterministic DNNs, and thus take a different route. Specifically, we introduce noise *only for quantifying* the MI between the input X and hidden layer L_i (as also suggested in [12]), but not into the DNN itself. Namely, we focus on minimizing the noise-regularized version of (1),

$$\mathcal{L}_{\text{IB}}^{\text{Reg}} = I(Y;L_i) - \beta I(X;L_i+\varepsilon), \tag{2}$$

where ε is Gaussian noise. As we show, the second term here can be interpreted as a weight decay penalty, which aligns with common practice in DNN training.

A key difficulty in using the MI as a loss function for training DNNs, is to obtain an accurate estimate of the MI, which is differentiable w.r.t. the net's parameters. To this end, we use an auxiliary net, similarly to the suggestion by Belghazi et al. [4], and in contrast to applying a variational lower-bound as in [2, 8]. However, as opposed to the original mutual information neural estimator (MINE) of [4], which fails to accurately estimate $I(X;L_i+\varepsilon)$ in our setting (see Sec. 3), we tailor the architecture of the auxiliary net specifically for our case in which L_i is a known deterministic function of X. This modification results in significantly more accurate estimates, as we show in Section 3.

2. Relation to weight decay

To understand the effect of replacing the penalty $I(X;L_i)$ by $I(X;L_i+\varepsilon)$, we can write

$$I(X;L_i(X)+\varepsilon) = h(L_i(X)+\varepsilon) - h(L_i(X)+\varepsilon|X), \tag{3}$$

where h denotes differential entropy, and we wrote $L_i(X)$ to emphasize that the latent representation is a deterministic function of the input X. First, notice that at zero noise level the second term in (3) becomes minus infinity (since $p_{L_i(X)|X}$ becomes a delta function), which highlights again the inadequacy of the penalty $I(X;L_i)$ for deterministic DNNs. Second, this term can be further simplified (for Gaussian noise) as

$$h(L_i(X)+\varepsilon|X) = \tfrac{1}{2}\log\left(|2\pi e \boldsymbol{\Sigma}_\varepsilon|\right), \tag{4}$$

where $\boldsymbol{\Sigma}_\varepsilon$ is the noise covariance matrix. Equation (4) is *independent* of the DNN parameters. This shows that the MI penalty $I(X;L_i(X)+\varepsilon)$ is in fact a penalty only on the entropy $h(L_i(X)+\varepsilon)$ of the representation L_i, which is a typical measure of compactness.

To develop further intuition, it is instructive to examine the simple case in which both the input X and the additive noise ε are (independent) Gaussian vectors and the transformation is linear, i.e. $L_i(X) = \boldsymbol{W}X + \boldsymbol{b}$. In this setting, simple calculation shows that

$$I(X;L_i+\varepsilon) = \frac{1}{2}\log\left(\left|\boldsymbol{W}\boldsymbol{\Sigma}_X\boldsymbol{W}^T + \boldsymbol{\Sigma}_\varepsilon\right|\right) + c, \tag{5}$$

where $\boldsymbol{\Sigma}_X$ is the input covariance matrix, and c is a term which is *independent* of $\boldsymbol{W}$ and $\boldsymbol{b}$. The expression in (5) attains its minimum when $\boldsymbol{W} = \boldsymbol{0}$, in which case X and L_i become independent. This highlights the fact that this term, in essence, induces a type of weight decay regularization. Clearly, the same intuition is also valid for non-linear settings, since weight decay (at the extreme) drives X and L_i to become independent and thus to have zero MI. Yet the precise form of the penalty is generally different between the MI regularizer and hand-crafted ones. Specifically, notice that the penalty in (5) is dependent on the input statistics (through the covariance matrix $\boldsymbol{\Sigma}_X$), as opposed to the popular ℓ_2 weight decay.

3. Accurate MI estimation

Given a DNN, to estimate the two MI terms in (2), we rely on the recently proposed MI neural estimator (MINE) of Belghazi et al. [4]. This method uses the fact that MI can be written in terms of the Kullback-Leibler divergence as $I(U;V) = \text{KL}(\mathbb{P}_{U,V}||\mathbb{P}_U \otimes \mathbb{P}_V)$, and exploits the dual representation of Donsker and Varadhan [6]

$$\begin{aligned} I(U;V) = \sup_D \Bigg[& \underset{(S,T)\sim\mathbb{P}_{U,V}}{\mathbb{E}}[D(S,T)] \\ & - \log\left(\underset{(S,T)\sim\mathbb{P}_U\otimes\mathbb{P}_V}{\mathbb{E}}[e^{D(S,T)}]\right)\Bigg], \end{aligned} \tag{6}$$

where the supremum is over all functions $D:\mathcal{U}\times\mathcal{V}\to\mathbb{R}$ for which the second expectation is finite. To approximate this value, the expectations are replaced by sample means over a training set, and the optimization is performed over a smaller family of functions -

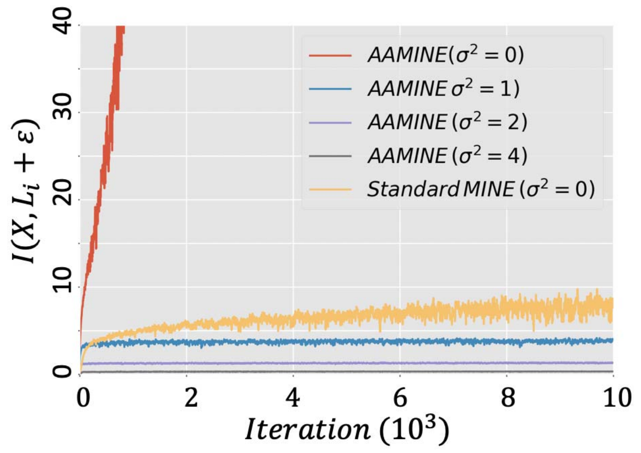

Figure 1: Estimating the (noise-regularized) MI between the input and last layer of a DNN using a standard MINE [4] and our AA-MINE (Sec. 3). For zero noise level ($\sigma^2 = 0$), the MI is theoretically infinite, which is captured by our AA-MINE but not by the standard MINE. When increasing the regularization, our AA-MINE is stable and estimates decreasing values of MI, as expected. Here the DNN is a 3-layer MLP with random weights.

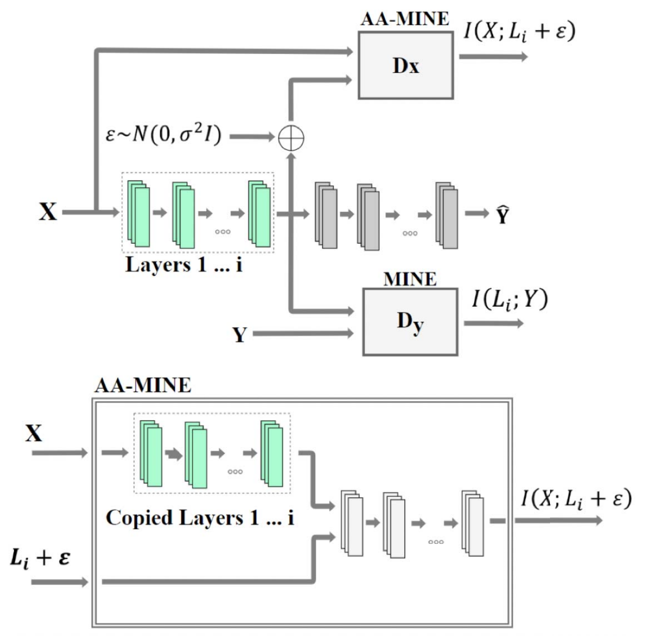

Figure 2: To measure the MI between the features at layer i and the target labels Y, we use the MINE estimator of [4]. To measure the MI between a noisy version of the features at layer i and the input X, we use our AA-MINE, which contains an internal copy of layers $1, \ldots, i$ (see zoom-in on the bottom).

those implementable by a deep net of predefined architecture. Notice that to optimize (6), D must learn to discriminate between (i) pairs (u, v) drawn from the joint distribution $\mathbb{P}_{U,V}$, and (ii) pairs (u, v) drawn *independently* from the marginal distributions $\mathbb{P}_U, \mathbb{P}_V$. This is similar to the principle underlying adversarial training [7].

In our setting, we estimate the term $I(Y; L_i)$ in (2) with a MINE, as described above. However, estimating the term $I(X; L_i + \varepsilon)$ in (2) is typically far more demanding, due to the higher dimensionality of X and the continuous nature of both arguments. Experimentally, we observed inconsistent and non-convergent behaviors for this estimate. In particular, as illustrated in Fig. 1, the estimate does not grow indefinitely as the noise variance is taken to 0, despite the fact that the true MI becomes infinite in this case (see Sec. 2). To improve the estimation accuracy, we exploit the strong prior knowledge we have in our setting, which is that the latent representation L_i is a known deterministic function of the input X. Specifically, $L_i = F_i(X)$, where F_i is the function implemented by the first i layers of the net. This implies that D is actually trying to discriminate between (i) pairs $(x_1, F_i(x_1) + \varepsilon)$ where x_1 is drawn from $\mathbb{P}_X$, and (ii) pairs $(x_1, F_i(x_2) + \varepsilon)$ where x_1, x_2 are *independently* drawn from $\mathbb{P}_X$. Intuitively, the easiest way to achieve this is by applying the function F_i on the first argument, and checking whether the result is close to the second argument, up to noise.

To assist the MINE in doing so, we implement a copy of the sub-net F_i within the discriminator D, allowing it to pass its first input through this sub-net (see Fig. 2). We coin this MI estimator the *architecture aware* MINE (AA-MINE). See the supplementary material for a detailed training algorithm of the AA-MINE. Notice that for the limit case of zero noise level, the discriminator need only check whether the two inputs are identical, which can be perfectly accomplished with a very simple architecture. This drives the optimization of the objective in (6) to infinity (see Fig. 1), which aligns with the theory.

4. IB based layer-by-layer training

The stage is now set for training a DNN layer-by-layer with the IB functional. Starting with the first hidden layer, our goal is to train each layer independently until convergence, freeze its weights, and then move on to the next layer. As our MI estimators are based on DNNs, we can achieve this goal by training them simultaneously with the layer of interest. Specifically, we write (2) using (6) and, as in adversarial training and in the original MINE work [7, 4], update the ith layer by alternating between updating the discriminators D_x, D_y (see Fig. 2) and updating the layer's parameters.

Once training is complete, it is convenient to seek an invertible transformation that brings the output $\hat{Y}$ closest to (the one-hot representation) Y. To achieve this, we *freeze the DNN parameters*, and post-train one additional fully-connected linear layer using a cross entropy loss. This determines the best linear transformation to bring $\hat{Y}$ closest to Y, without changing the MI between them (an invertible transformation does not change MI). This does not expand the net's capacity, since we do not use a nonlinear activation between the last layer of the net and the post-trained linear layer, so they can be combined into a single linear layer.

With this scheme, we trained a three-layered MLP on the MNIST dataset [10]. The resulting classification accuracies are reported in Table 1. Both when using 100% of the training set and when using only 1% of the training set, the performance of this training scheme is comparable to the baseline of training with the cross-entropy loss, and even slightly better. Also, notice that the penalty $I(X; L_i + \varepsilon)$ is essential for generalization, especially when using only 1% of the training set, and without it, there is a drop in test accuracy. This is thus the first experiment to directly validate the effectiveness of the IB principle for DNNs.

We also test the IB principle with a deep conv-net on the CIFAR-10 dataset [9] (see Table 1). Training with the full IB functional leads to better accuracy compared to the cross-entropy baseline, and again, the regularization term which promotes a compressed latent representation proves advantageous. An analysis of the information plane dynamics during training and all training details appear in the supplementary material.

Figure 3 shows the t-SNE embeddings [11] of the latent representations L_i of our MNIST net. Recall that the regularization term $I(X; L_i + \varepsilon)$ enforces reduced entropy of the latent representation L_i (see Section 2). For large β values, the representation becomes quantized, which is a form of compression and indeed reduces the differential entropy by decreasing the effective support of the representation[2].

[2]Differential entropy, in contrast to (discrete) entropy, depends not only on the number of clusters but also on their sizes. For example, the entropy of a *single* Gaussian cluster is $\frac{1}{2}\log(|2\pi e\Sigma|)$, which can be arbitrarily large. The entropy of a continuous distribution becomes smaller as its (effective) support becomes smaller, eventually tending to minus infinity for a distribution that is supported on a finite discrete set of points.

Data Set (%)	Training method	Test acc. (%)
MNIST 1%	Cross-Entropy	85.78
	$\mathcal{L} = I(Y; L_i)$	85.12
	$\mathcal{L} = I(Y; L_i) - \beta I(X; L_i + \varepsilon)$	**86.57**
MNIST 100%	Cross-Entropy	97.73
	$\mathcal{L} = I(Y; L_i)$	97.77
	$\mathcal{L} = I(Y; L_i) - \beta I(X; L_i + \varepsilon)$	**98.09**
CIFAR-10 100%	Cross-Entropy	58.23
	$\mathcal{L} = I(Y; L_i)$	60.59
	$\mathcal{L} = I(Y; L_i) - \beta I(X; L_i + \varepsilon)$	**61.75**

Table 1: Test-set accuracy on the MNIST and CIFAR-10 datasets for layer-by-layer training with the IB functional (2). Pushing each layer towards the optimal IB curve results in comparable performance to the cross-entropy objective baseline. Notice that maximizing the MI with the output $I(Y; L_i)$ alone leads to inferior generalization, showing the importance of the regularization term $I(X; L_i + \varepsilon)$.

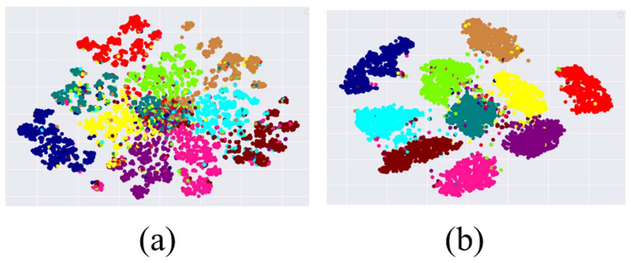

Figure 3: Visualizing the effect of training using the IB principle with t-SNE embeddings. Here we show the embedding of the first layer with a trade-off coefficient of (a) $\beta = 10^2$, and (b) $\beta = 10^{-3}$. Increasing β leads to a sort of discretization of the latent representation, which indeed acts as a form of compression.

5. Conclusion

Our experiments demonstrate that training DNNs with the IB functional leads to competitive prediction performance. This provides strong and direct empirical evidence for the validity of the IB theory of deep learning. Our training scheme was made possible by two key changes to prior attempts. First, we used a noise-regularized version of the IB functional, which removes the theoretical difficulty of the MI between layers being infinite, while still being consistent with the intuition of a complexity penalty. Second, we derived an MI estimation scheme tailored for MI estimation between deterministic DNN layers, which is capable of *accurate* estimation in a scenario which is generally intractable.

Acknowledgement This research was supported by the Ollendorff Minerva Center of the Technion.

References

[1] Alessandro Achille and Stefano Soatto. Information dropout: Learning optimal representations through noisy computation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2018.

[2] Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. Deep variational information bottleneck. In *International Conference on Learning Representations (ICLR)*, 2018.

[3] Rana Ali Amjad and Bernhard C Geiger. Learning representations for neural network-based classification using the information bottleneck principle. *arXiv preprint arXiv:1802.09766*, 2018.

[4] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Devon Hjelm, and Aaron Courville. Mutual information neural estimation. In *International Conference on Machine Learning (ICML)*, pages 530–539, 2018.

[5] Yochai Blau and Tomer Michaeli. Rethinking lossy compression: The rate-distortion-perception tradeoff. In *International Conference on Machine Learning (ICML)*, pages 675–685, 2019.

[6] Monroe D Donsker and SR Srinivasa Varadhan. Asymptotic evaluation of certain markov process expectations for large time. iv. *Communications on Pure and Applied Mathematics*, 36(2):183–212, 1983.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems (NIPS)*, pages 2672–2680, 2014.

[8] Artemy Kolchinsky, Brendan D Tracey, and David H Wolpert. Nonlinear information bottleneck. *arXiv preprint arXiv:1705.02436*, 2017.

[9] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, 2009.

[10] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[11] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research (JMLR)*, 9(Nov):2579–2605, 2008.

[12] 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 (ICLR)*, 2018.

[13] Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*, 2017.

[14] Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. In *The 37th annual Allerton Conference on Communication, Control, and Computing*, pages 268–377, 1999.

[15] Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *IEEE Information Theory Workshop (ITW)*, pages 1–5, 2015.