Politecnico di Torino

Master in Physics of Complex Systems

A. a. 2022/2023

graduation session 07/2023

Plasticity across neural hierarchies in artificial neural network

Relatori:
prof. Matteo Marsili
prof. Andrea Pagnani

Candidato:
Carlo Orientale Caputo
Matr. *s*302914

Contents

1 Introduction

Deep neural network can extract a hierarchy of relevant features from the data, that can be used both for classification and generation task [8][7]. These models have state-of-the-art performance in object/speech recognition and language translation [6]. However, many characteristics of the way the network processes the information and the reason why they work so well is still unclear. In this work we analyze some features of a deep belief network during training across different layers in an unsupervised setting.

First of all, we study how the plasticity varies across the network's layers. To do this we compute the variation of the architecture's weights when the dataset to be learned is changed. We observe an increasing behaviour of the plasticity across layers, meaning that the features learned in deeper layers are more dataset dependent, instead the shallow ones are more generic.

Then we analyze some features of the internal representation of the hidden layers, that is the probability distribution learned over the hidden nodes of each layer. We find that shallow layers are well described by a pairwise model, while in deep layers, higher order interactions seem to be more present. This could be related with the hierarchical extraction of features performed by the

network.

Finally, we observe that the representations across the layers become close to the hierarchical feature model [9], a theoretical model describing the internal representation of a learning machine that is consistent with the principles of maximal a priori ignorance and of maximal relevance, that depends on 1 parameter.

2 Plasticity and Internal Representation

2.1 Deep belief network

A deep belief networks (DBN) consists of restricted boltzmann machines (RBM) stacked one on top of the other. Each RBM is a Markov random field with pairwise interactions defined on a bipartite graph of two non interacting layers of variables: visible variables $\mathbf{v} = (v_1, .., v_N)$ representing the data, and hidden variables $\mathbf{h} = (h_1, ..., h_M)$ that are the latent representation of the data. The mesure of a single RBM is (repeated indices are summed):

$$p(\mathbf{x}, \mathbf{h}|\mathbf{W}, \mathbf{c}, \mathbf{b}) = \frac{1}{Z} \exp\left(W_{ij} x_i h_j + x_k c_k + h_l b_l\right). \tag{1}$$

The entire DBN has the top two layers with undirected connections between them while lower layers receive top-down, directed connections from the layer above (see figure 1).

The difference between directed and undirected graphical model of random variables relies on the factorization of the joint distribution of the nodes and in the way we can sample from it. In an undirected model the full probability distribution can be factorized over the cliques (a fully connected subset of nodes) of the graph. Given a set of random variables $\mathbf{X} = (X_1, .., X_n)$ represented with an undirected graphical model G:

$$p(\mathbf{X}) = \prod_{C \in cl(G)} \phi_C(\mathbf{X}_C) \tag{2}$$

where $cl(G)$ is the set of cliques of G, and $\phi_C(\mathbf{X}_C)$ are functions that depends only on the set of nodes that belong to the clique C. In a directed model $G = (V, E)$, each edge E is represented with an arrow, and for each node $v \in V$ we can define the set of parents of v: $pa(v)$ as those vertices pointing directly to v. The full probability distribution can be factorized as product of single node marginal, conditioned on the parents node:

$$p(V) = \prod_{v \in V} p(v|pa(v)). \tag{3}$$

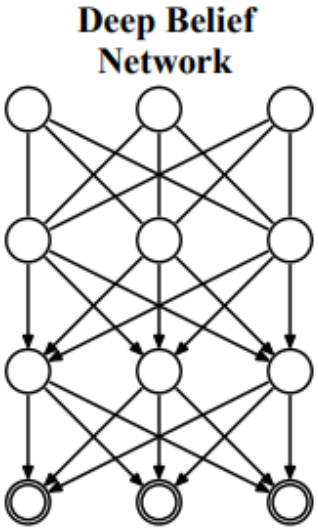

Figure 1: A three layer Deep Belief Network

In our case, the practical consequences of deal with a directed model is in the way we sample from it. To obtain a sample from a DBN we can use Gibbs sampling to sample the equilibrium of the RBM on the top $p_{RBM}(\mathbf{h}_n, \mathbf{h}_{n-1})$, and then using this data to sample from the hidden nodes directly connected to $\mathbf{h}_{n-1}$ using only the marginal $p(\mathbf{h}_{n-2}|\mathbf{h}_{n-1})$, then propagate the signal till the visible layer using the marginal distributions $p(\mathbf{h}_l|\mathbf{h}_{l-1})$. Instead if it were an undirected model (deep Boltzmann machine), to sample from an intermediate layer we would have needed the signal from both the top and the down layer connected to the intermediate one.

The reason why our architecture is a DBN is a consequence of the way we train it. We learn the weights one layer at a time, following the prescription of Hinton [5]. It consists of training the first RBM on the data, then propagate the input $\{\mathbf{v}^\mu\}_{\mu=1}^L$ data forward to the first hidden layer, obtain the hidden states $\{\mathbf{h}^\mu\}_{\mu=1}^L$ and use them as input for training the second hidden layer. This type of training procedure was proven [5] to increase a variational lower bound for the log likelihood of the data set (more information about the architecture and the training procedure are given in A.1). So at the first step we learn a RBM: $p(\mathbf{v}, \mathbf{h}_1) = p(\mathbf{v}|\mathbf{h}_1)p(\mathbf{h}_1)$, then we substitute the model for $p(\mathbf{h}_1) = \sum_{\mathbf{v}} p(\mathbf{v}, \mathbf{h}_1)$ with another model coming from a second RBM trained on a sample from $p(\mathbf{h}_1)$, obtaining in the end:

$$p(\mathbf{v}, \mathbf{h}_1, \mathbf{h}_2) = p(\mathbf{v}|\mathbf{h}_1)p(\mathbf{h}_1, \mathbf{h}_2). \tag{4}$$

Then repeating this procedure till layer n. So now to sample from an inter-

mediate layer we just need the signal coming from the layer above.

2.2 Plasticity of the architecture

To study the plasticity in function of the layers of this architecture we first train it using hand-written digits data set (MNIST) and learn a set of parameters $\{\mathbf{W}_1^\mu\}_{\mu=1}^l$ for each layer. Then we use the same architecture to learn Zalando's article images (fashion MNIST) and obtain some new parameters $\{\mathbf{W}_2^\mu\}_{\mu=1}^l$ (using $\{\mathbf{W}_1^\mu\}_{\mu=1}^l$ as initial condition of the training). Finally, we compute the L^2 norm of the difference between each set of parameters for each layer, and in figure 2 you can see the results also for DBN trained first with FMNIST and then with MNIST, and from EMNIST (hand-written letters) to MNIST.

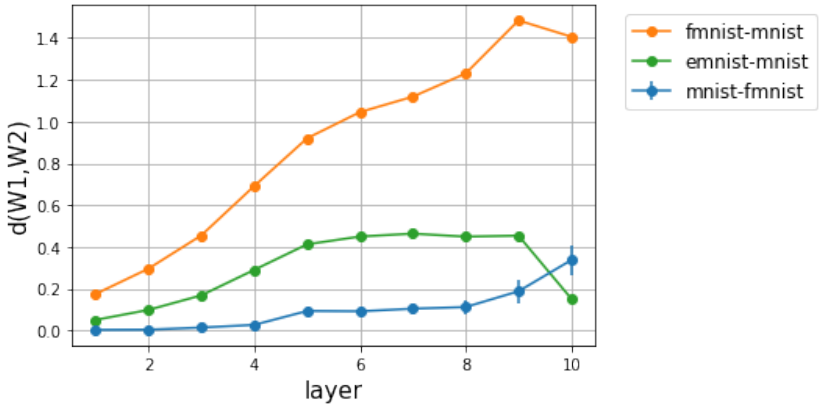

Figure 2: difference of the weights in function of layer after training the same architecture with two different data sets. The error bars on mnist-fmnist curve were calculated from 10 simulations.

We can see how shallow layers' weights change less with respect to deep layers. Probably the weights of the first layers are very generic, they do not capture specific features of the dataset. Instead the deep layers have a more specific representations, and the weights are more data dependent. This result seems coherent with the observation that convolutional deep belief network can extract a hierarchical representation of data: learning some oriented, localized edge filter for the first layers' weights, while more high level feature were learned by the deep layers [7].

2.3 Feature of Internal Representation: order of interactions

The internal representation of the network is the probability distribution learned over the hidden layers: $p_l(\mathbf{h})$. Loosely speaking, it gives us information on how the network organize the feature space. An interesting observable that can be measured, to understand some features of the internal representations, is the pairwise-ness. This may be defined as the Kullback-Leibler distance between the internal representation and the best pairwise model: $DKL(p_l||p_l^{(2)})$, where:

$$p_l^{(2)}(\sigma) = \frac{1}{Z} \exp\left(\sum_{i<j} J_{ij}^l \sigma_i \sigma_j + \sum_i h_i^l \sigma_i\right) \tag{5}$$

and Z is the partition function and the parameters J and h are estimated using maximum likelihood (more information are given in A.2). The results of the DKL for different layers is showed in figure 3, for 3 different DBN trained with MNIST, FMNIST and EMNSIT dataset.

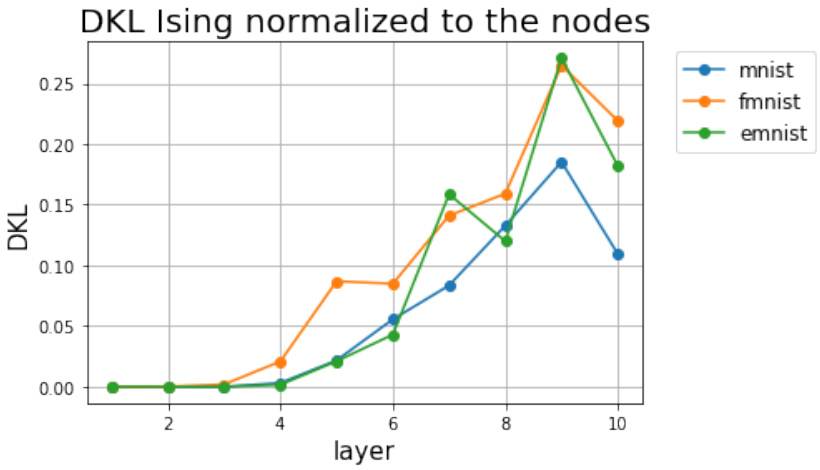

Figure 3: DKL between the internal representation of each layer and the best pairwise model describing that representation, normalized with the number of nodes of each layer. It is estimated from the sample $\{\mathbf{h}_l^\mu\}_{\mu=1}^N$. The DBN was trained with mnist, fmnsit and emnist dataset.

First layers are more dominated by pairwise statistics than deep layers, and this could be related to the hierarchical feature extraction done by the deep network. The features learned in the first layers are simple localized

and oriented edges, with low correlations between them. Instead the representations of the deep layers carry information about high level features -obtained as a combination of the simple one- and the resulting internal representation has a rich dependencies among the nodes. This doesn't mean that the deep layers are described by more complex model (in terms of the number of parameters), but only that higher order interactions are more present. In fact, as we will see, deep layers are better described by a simple model, with just 1 parameter, containing all order of interactions. Instead, shallow layers are better described by a more complex (with more parameters) pairwise model. This seems also coherent with the behavior of the intrinsic dimensionality of the internal representation in function of layer of a DNN showed in [2], suggesting that more complex models are learned in shallow layers' representations.

The information about the order of interaction in $p_l(\mathbf{h})$ can be obtained in another way. First, we need to translate binary variable $h_i = 1, 0$ into spin variable $\sigma = 2s_i - 1$. Next, we can use that every function $p : \{-1, 1\}^N \to R$ can be decompose in the following way:

$$p(\sigma) = \frac{1}{Z} e^{\sum_\mu g^\mu \phi^\mu(\sigma)} \tag{6}$$

with:

$$\phi^\mu = \prod_{i \in \mu} \sigma_i \tag{7}$$

and we can compute g^μ as:

$$g^\mu = 2^{-n} \sum_\sigma \phi^\mu(\sigma) \log(p(\sigma)). \tag{8}$$

We can estimate it using our sample $\{\mathbf{h}_l^\mu\}_{\mu=1}^N$ for each layer.

The strength of a certain order k of interaction for a given layer can be calculated as:

$$G_k^l = \sqrt{\frac{1}{\binom{n}{k}} \sum_{\mu:|\mu|=k} (g_l^\mu)^2} \tag{9}$$

with n the number of nodes of that hidden layer. In figure 4 you can see G_k^l in function of layer for different order of interactions. Because these quantities are estimated using a sample, for wide layers they have large variance, so to deal with this problem we calculate them over 10 randomly selected nodes, for 20 times and take an average.

Consistently with what observed using the DKL with the Ising model, higher order statistics are more present in deep layers.

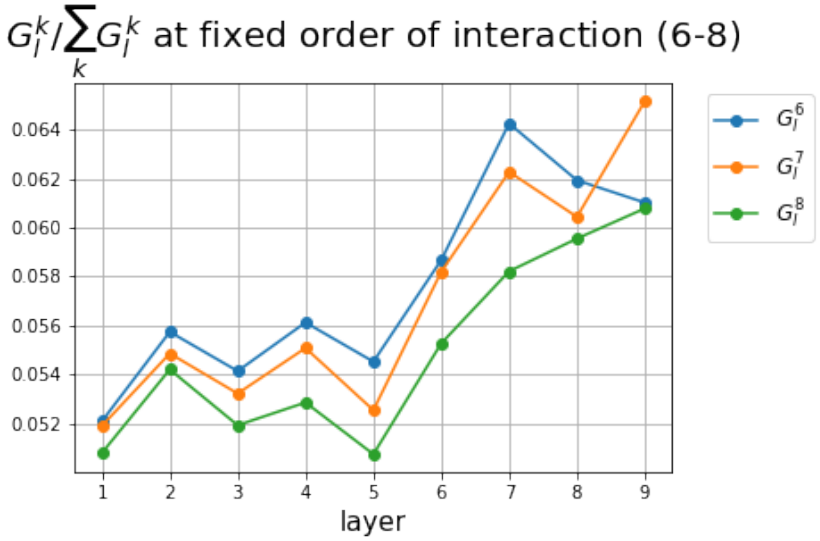

Figure 4: Strength of the normalized conjugate parameters, in function of layers for different orders of interactions. We calculate them over 10 randomly selected spins per each layer, for 20 times and then we take an average.

3 Emergence of hierarchical structure

3.1 Hierarchical feature model

The hierarchical feature model (HFM) is a model for the internal representation of a system. It is consistent with the principles of maximal a priori ignorance (if the latent representation $\mathbf{h}$ has feature k, no information on whether feature $i < k$ is present or not is available, a priori: $p(h_i|h_k = 1)$ is uniform over h_i) and maximal relevance, defined as the entropy of the frequency distribution [9].

Given a sample $\{\mathbf{h}_{\mu=1}^N\}$ from the internal representation of a layer, the probability of the frequency is: $p(k) = m_k k/N$ where m_k is the degenaracy of frequency k, then the relavance is defined as:

$$H[k] = \sum_k \frac{m_k k}{N} \log\left(\frac{m_k k}{N}\right). \tag{10}$$

In a work by prof. Marsili et al. [3] this quantity is argued to provide a quantitative measure to the useful information content that a certain representation has on its own generative process. It is also argued that maximally informative representations should maximize the relevance at fixed level of resolution (defined as the entropy $H[\mathbf{h}]$ of the sample $\{\mathbf{h}_{\mu=1}^N\}$). Another

work [8] showed that the internal representation of this architecture seems to respect this principle of maximal relevance.

The principle of a priori ignorance implies that $p(\mathbf{h})$ must be a function of the largest index i for which $h_i = 1$, i.e. it must be a function of

$$m_{\mathbf{h}} = max\{i : h_i = 1\}. \tag{11}$$

The principle of maximal relevance implies that this function must be an exponential, as shown in [3], so:

$$p(\mathbf{h}) = \frac{1}{Z} e^{-g\mathcal{H}(\mathbf{h})}, \qquad \mathcal{H}(\mathbf{h}) = max\{m_{\mathbf{h}} - 1, 0\} \tag{12}$$

where the partition function is:

$$Z = \sum_{\mathbf{h}} e^{-g\mathcal{H}(\mathbf{h})} = 1 + \frac{\xi^n - 1}{\xi - 1}, \qquad \xi = 2e^{-g}. \tag{13}$$

In the limit $n \to \infty$ the model exhibits a phase transition at $g_c = \log 2$, signalled by a divergence of the variance of the energy (specific heat). For value of $g < g_c$ the model describes a representation that spans an extensive number of features (the $\mathrm{E}_{\mathbf{s}}[\mathcal{H}]$ is extensive), while for $g > g_c$ the $\mathrm{E}_{\mathbf{s}}[\mathcal{H}]$ is of order one.

This model describes an internal representation that learns a hierarchy of features: the first features are more generic and can describe many different elements in the dataset, while later features add more specific details to the representation. Indeed both the magnetization $\langle h_i \rangle$ and the single node entropy $H[h_i]$ are decreasing along the hierarchy.

3.2 Feature of Internal Representation: Hierarchical structure

To see if the internal representation of this DBN has an organization of feature space similar to the HFM, we calculate the single node magnetization and entropy of the internal representation of a DBN trained on MNIST dataset, and we sort them in decreasing order. The results are in figure 5 and 6, you can see how these quantities decrease along the hierarchy as in the HFM, however we will see that only the deep layers' hierarchy is close to the HFM.

First, we calculate the parameter g^* of the HFM that best fits the data of a particular layer. Because the HFM is an energy based model, we just

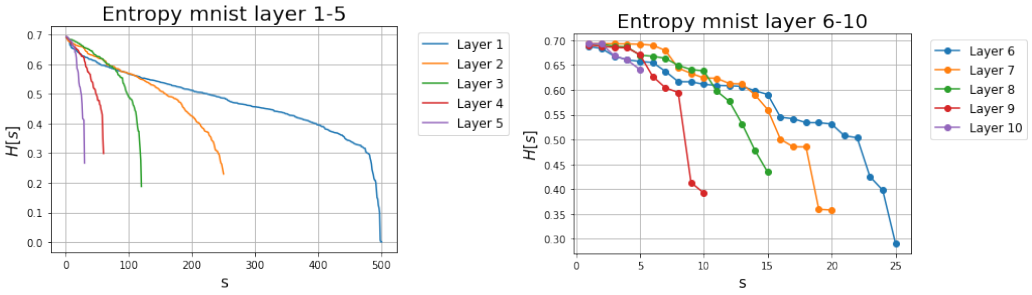

Figure 5: $H_l[h_i]$ in function of the nodes for each layer. The nodes are sorted so that the entropy is decreasing.

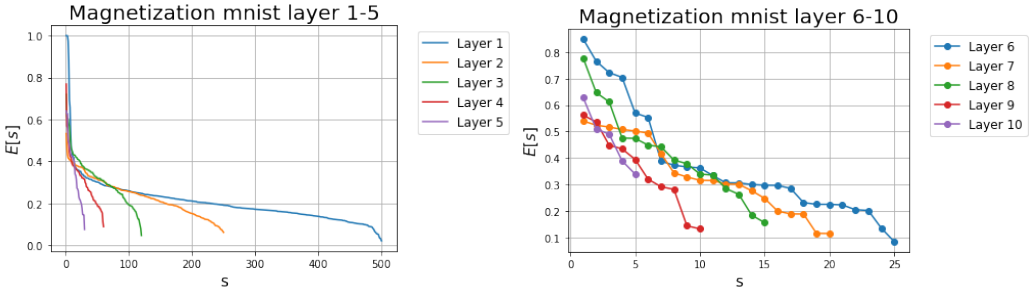

Figure 6: $\langle h_i \rangle$ in function of the nodes for each layer. The nodes are sorted so that the magnetization is decreasing.

need to find the g^* such that the energy average over the model matches the empirical one:

$$\langle \mathcal{H}(\mathbf{h}) \rangle_{\mathcal{D}} \equiv \frac{1}{M} \sum_{n=1}^{M} max\{m_{\mathbf{h}_n} - 1, 0\} = \sum_{\mathbf{h} \in \mathbf{S}} max\{m_{\mathbf{h}} - 1, 0\} P_{HFM}(\mathbf{h}) \equiv \langle \mathcal{H}(\mathbf{h}) \rangle_{P_{HFM}}. \tag{14}$$

Knowing that the average of a HFM's energy of n nodes is:

$$\langle \mathcal{H}(\mathbf{h}) \rangle_{P_{HFM}} = \xi \left(\frac{n\xi^n - 1}{\xi^n + -2} - \frac{1}{\xi - 1} \right), \qquad \xi = 2e^{-g}. \tag{15}$$

In figure 7 you can see the value of g^* for each layer of the DBN trained with mnist.

Then, we calculated the magnetization in function of the node position in the hierarchy, for a HFM with a parameter equal to the g^* of figure 7 that best fits each DBN's layer. The formula for the magnetization is easy to obtain

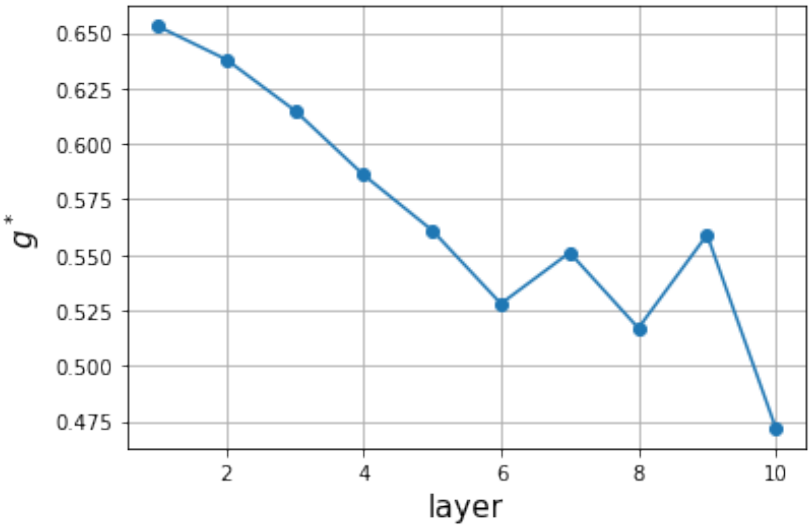

Figure 7: g^* that solves equation 14 for the activations of each layer of a DBN trained with mnist dataset.

conditioning on the energy. For this purpose let's define $\mathbf{S}_k = \{\mathbf{s} : m_\mathbf{s} = k\}$ the set of stats with the last spin up being the k^{th} one, then:

$$E[s_i] = p(s_i = 1) = \sum_{k=0}^{n} p(s_i = 1|\mathbf{s} \in \mathbf{S}_k)p(\mathbf{s} \in \mathbf{S}_k) = \frac{\xi^n + \xi^i - 2\xi^{i-1}}{2(\xi^n + \xi - 2)}. \quad (16)$$

If we compare these magnetizations with the one that we found for the DBN's layers, we can see that only deep layers are well described by the HFM. You can see in figure 8 and 9 these plots for the first two and the last two layers.

Another feature of the internal representation learned by the DBN, that is not immediately clear from figure 7, is that deep layers' representations are getting closer to the one of a HFM at the critical value $g_c = \log 2$. From a coding theory prospective, it is an efficient way of using the feature, because the number of bits we need to code for a data grows linearly with its number of feature. At criticality the probability of having the last feature at position m $p(m)$ is uniform: $-\log p(m) = -(m-1)\log 2 - \log p(\mathbf{s}|m) = const.$ So $-\log p(\mathbf{s}|m) \sim (m-1)\log 2$. You need $m-1$ bits to describe an object with m feature, (that is precisely the number of bits you need to distinguish it from the other 2^{m-1} different objects with the same number of features).

To see why deep layers are more critical we need to remember that the max likelihood estimator $T(g)$ of the parameter g has its own variance. For Cramer-Rao bound we know that the variance of this estimator has as lower

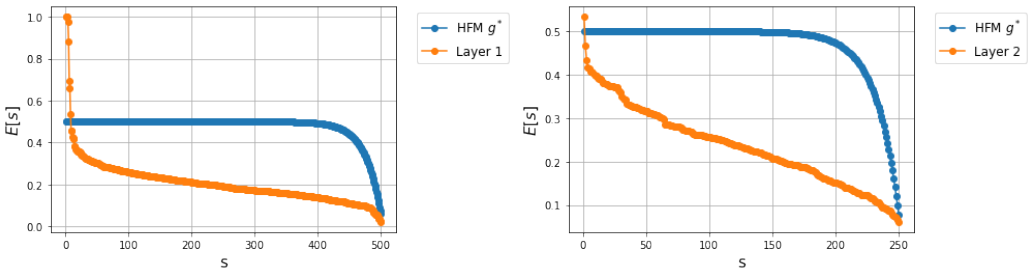

Figure 8: magnetization in function of nodes for the first two layers of a DBN trained with mnist dataset and the corresponding magnetization of the HFM with the parameter g that best fits the layers' activations. They are not well described by a HFM.

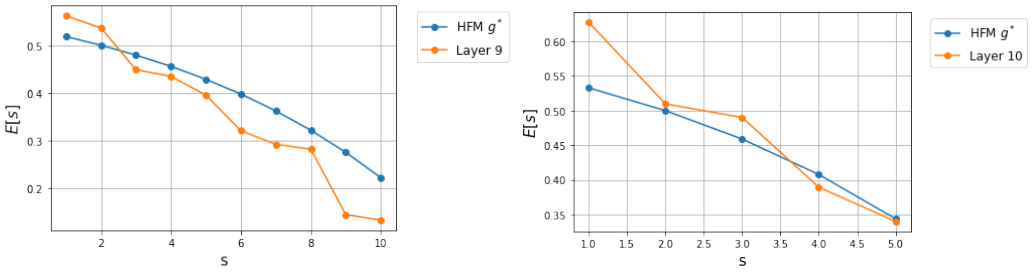

Figure 9: magnetization in function of nodes for the last two layers of a DBN trained with mnist dataset and the corresponding magnetization of the HFM with the parameter g that best fits the layers' activations. They are better described by a HFM than shallow layers.

bound the inverse of the Fisher information:

$$\mathbf{V}[T(g)] \geq \frac{1}{MJ(g)} \tag{17}$$

where M is the number of samples used to estimate g and $J(g)$ is the Fisher information, defined as:

$$J(g) = \int d\mathbf{s} p(\mathbf{s}|g) \left[\frac{\partial}{\partial g} p(\mathbf{s}|g)\right]^2. \tag{18}$$

For an exponential family the Fisher information of g_c is the variance of the conjugate observable, in our case for a HFM with n nodes:

$$J(g_c) = \mathbf{V}[\mathcal{H}_{g_c}] = \frac{n(n-1)[n(n+5)-2]}{12(n+1)^2}. \tag{19}$$

So to see how really each layer representation is close to criticality we can plot the ratio between the deviation from the critical value and the standard deviation around it in function of layer: $(g_c - g^*)/\delta g = \sqrt{\langle \delta E^2 \rangle}(g_c - g^*)$. As you can see in figure 10, this quantity decreases over layer suggesting that the model learned in deep layers is getting close to the critical HFM.

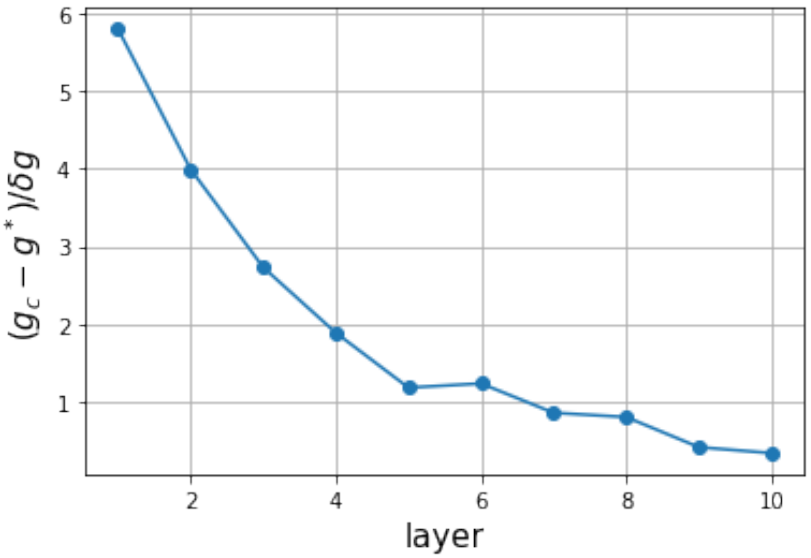

Figure 10: Ratio between the deviation from the critical value g_c and the standard deviation of the estimator in function of layers. Again g^* is the value that solves equation 14 for each layer of a DBN trained with mnist dataset.

Finally, to measure how much the internal representation of each layer is close to the hierarchical feature model, we compute the $DKL(p_l||p_{HFM})$ between $p_l(\mathbf{h})$ and the HFM $p_{HFM}(\mathbf{h})$ that best fits the data of each layer (again this is estimated using a sample $\{\mathbf{h}_{\mu=1}^N\}$). In figure 11 you can see the results for 3 DBN trained on MNIST, FMNIST and EMNIST dataset. In all these cases the DKL decreases along the layers, again suggesting that representations in deep layers are getting closer to this simple model (with just 1 parameter), but with a richer structure.

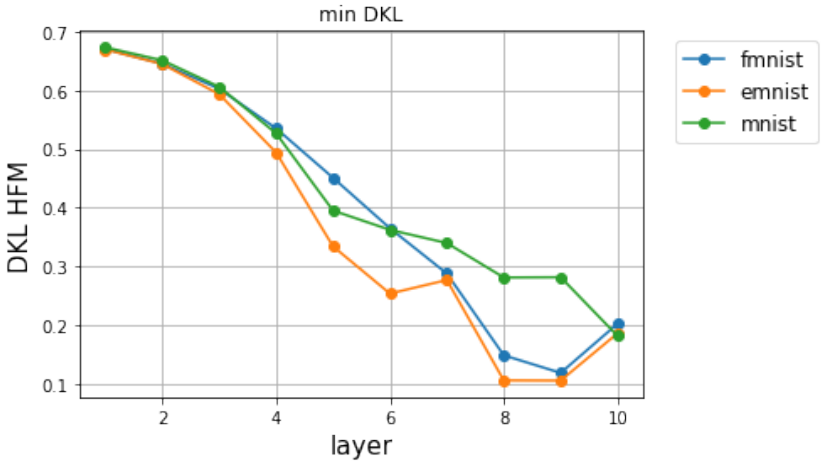

Figure 11: DKL between the internal representation of each layer and the best-fit HFM, normalized to the number of nodes of each layer for 3 different datasets.

4 Conclusion

The main results that we obtain in this work concern the behavior of the plasticity across different layers of a DBN in an unsupervised setting, and the characterization of some features of the internal representations of different DBN's layers.

First, we observe an increasing behaviour of the plasticity (defined as the amount of change of the weights when the dataset to be learned is changed) across the layers. It suggests that the features learned in the shallow layers are more generic and simple, and for this reason they are able to describe different dataset without changing much. Instead, the high level features, learned in deep layers, capture more specific characteristics of a particular dataset. This is coherent with what was found in [7].

Then, we analyze the internal representation of the DBN after training with different datasets and what we observe is that shallow layers' representations can be described by a pairwise model. It suggests that the simple feature of the data, learned in the firsts layers, are just pairwise correlated. Instead deep layers, that contain information on high level feature of the data, seem to learn representation where high order interactions are present. Furthermore, these representations become close to a theoretical model (HFM) describing an organization of feature space consistent with the principles of maximal a

priori ignorance and maximal relevance.

For future works it would be interesting to see how general are these conclusions about the plasticity and the internal representation -for example trying to extend them to different deep neural networks like convolutional neural network in an unsupervised setting, or variational autoencoder.
It is also interesting to see whether these models are able to describe the way in which certain brain areas (e.g. the ventral stream) elaborate external stimuli and learn from them. Searching similarities between the structure, the internal representation and the plasticity of both the natural and the artificial architecture could be a starting point to find some common principles in the way they elaborate the information, in order to perform complex cognitive function.

A Simulation details

A.1 Training of DBN

The DBN used in our experiment has a visible layer with 784 nodes and 10 hidden layers with the following number of nodes: 500-250-120-60-30-25-20-15-10-5. To train this architecture we used the algorithm proposed by Hinton [5]. It consists in training the architecture layer by layer, using the hidden layer of an RBM as the visible of the following RBM. In this way, we can learn the weights one layer at a time with a guarantee of improving a variational lower bound of the log likelihood of the data under the full generative model, as was shown in [5]. This allows us to use approximated training methods like contrastive divergence (CD) and still being able to obtain a good generative model.

To learn the parameters of a single RBM we used a stochastic gradient ascent of the log-likelihood of the training datasets $\mathcal{D} = \{\mathbf{v}^1, ..., \mathbf{v}^M\}$. The measure of an RBM is given by equation 1 and the log-likelihood is defined as:

$$\begin{aligned} \mathcal{L}(\mathbf{W}, \mathbf{b}, \mathbf{c}|\mathcal{D}) &= \frac{1}{M}\sum_{m=1}^{M} \ln p(\mathbf{v} = \mathbf{v}^m|\mathbf{W}, \mathbf{b}, \mathbf{c}) \\ &= \frac{1}{M}\sum_{m=1}^{M} \ln \sum_{\{\mathbf{h}\}} e^{-E(\mathbf{v}^m, \mathbf{h}|\mathbf{W}, \mathbf{b}, \mathbf{c})} - \ln Z \end{aligned} \tag{20}$$

knowing the energy to be:

$$E(\mathbf{v}, \mathbf{h}|\mathbf{W}, \mathbf{b}, \mathbf{c}) = -\sum_{ij} v_i W_{ij} h_j - \sum_i c_i v_i - \sum_k b_k h_k \tag{21}$$

the components of the gradient are:

$$\frac{\partial \mathcal{L}}{\partial W_{ij}} = \langle v_i h_j \rangle_{\mathcal{D}} - \langle v_i h_j \rangle_p \tag{22}$$

$$\frac{\partial \mathcal{L}}{\partial b_k} = \langle h_k \rangle_{\mathcal{D}} - \langle h_k \rangle_p \tag{23}$$

$$\frac{\partial \mathcal{L}}{\partial c_i} = \langle v_i \rangle_{\mathcal{D}} - \langle v_i \rangle_p \tag{24}$$

where $\langle f(\mathbf{v}, \mathbf{h}) \rangle_{\mathcal{D}} = M^{-1} \sum_m \sum_{\mathbf{h}} f(\mathbf{v}^m, \mathbf{h}) p(\mathbf{h}|\mathbf{v}^m)$ is the average over the dataset, and $\langle f(\mathbf{v}, \mathbf{h}) \rangle_p$ is the average over the measure of equation 1.

The components of the gradient are computed at each epoch over a batch of size 64, to introduce stochasticity and reduce the likelihood of being trapped in local minima. In theory, one can evaluate the average over the model $\langle f(\mathbf{v}, \mathbf{h}) \rangle_p$ using parallel Monte Carlo Markov chains (MCMC), with a large number of steps to ensure a sampling of the equilibrium distribution for each epoch. In practice some approximation schemes are used. For example in Contrastive Divergence-k (CD-k), the Markov chains are initialized inside the batch used to compute the gradient and k Monte Carlo steps are performed -the idea is that the dataset is a good approximation of the equilibrium samples of a well trained RBM. In Persistent Contrastive Divergence-k (PCD-k) the MCMC is initialized in the configuration of the previous epoch, with a random initialization for the first epoch -the idea is that if the parameters of the distribution change slowly, than also the equilibrium distribution doesn't change much, so the chain of the previous step could be a good initialization for the current epoch.

The consequences of these approximations are not well understood. From an analytical point of view, Decelle et al. [4] [1] proved that the equilibrium distribution learned by an RBM trained with CD-10 was not able to capture the statistics of the dataset, but it can be a good generative model if it were sampled out of equilibrium. Instead they observed that PCD-10 was able to learn a good equilibrium distribution.

To the best of our knowledge we don't have analytical proof of the consequences of these approximation scheme in deep architectures, however

the layer by layer training improve the full generative model [5], and the results of our experiments are independent from the approximation scheme used.

A.2 Boltzmann learning of Ising model

The Ising model can be seen as a max entropy model, where one looks for the most uniform probability distribution that satisfies some constraints. The constraints are such that, given a set of observables $f_i(\mathbf{s}) : \mathbf{S} \to R$, with $i = 1, .., p$, their model average needs to match the empirical one:

$$\langle f_i \rangle_{\mathcal{D}} \equiv \frac{1}{M} \sum_{n=1}^{M} f_i(\mathbf{s}_n) = \sum_{\mathbf{s} \in \mathbf{S}} f_i(\mathbf{s}) P(\mathbf{s}) \equiv \langle f_i \rangle_P \tag{25}$$

with M the number of data. If we choose as observables the single node magnetization $f_i(\mathbf{s}) = s_i$ and the two points correlation $f_{ij}(\mathbf{s}) = s_i s_j$, we can introducee a set of Lagrange multipliers $\{h_i, J_{i,j}\}$ to enforce each constraints, and the resulting max entropy model will be an Ising model:

$$p(\mathbf{s}|\mathbf{h}, \mathbf{J}) = \frac{1}{Z} \exp \sum_{i,j} J_{ij} s_i s_j + \sum_i h_i s_i. \tag{26}$$

It is known that, for an exponential family, finding the Lagrange multipliers that satisfied the constraints is the same as maximizing the log-likelihood $\mathcal{L}(\mathbf{J}, \mathbf{h}|\mathcal{D})$ of the empirical data, whose gradient components are:

$$\frac{\partial \mathcal{L}}{\partial J_{ij}} = \langle s_i s_j \rangle_{\mathcal{D}} - \langle s_i s_j \rangle_p \tag{27}$$

$$\frac{\partial \mathcal{L}}{\partial h_i} = \langle s_i \rangle_{\mathcal{D}} - \langle s_i \rangle_p . \tag{28}$$

To find the parameters we perform a gradient ascent of the log likelihood. We use PCD to evaluate the average over the model, using 64 parallel Markov chain of length $10 \cdot n$, with n the total number of spins in the chain, for $\sim 10^3$ epochs.

References

[1] Elisabeth Agoritsas et al. "Explaining the effects of non-convergent sampling in the training of Energy-Based Models". In: *arXiv e-prints*, arXiv:2301.09428 (Jan. 2023), arXiv:2301.09428. DOI: 10.48550/arXiv.2301.09428. arXiv: 2301.09428 [cs.LG].

[2] Alessio Ansuini et al. *Intrinsic dimension of data representations in deep neural networks*. 2019. arXiv: 1905.12784 [cs.LG].

[3] Ryan John Cubero et al. "Statistical criticality arises in most informative representations". In: *Journal of Statistical Mechanics: Theory and Experiment* 2019.6 (June 2019), p. 063402. DOI: 10.1088/1742-5468/ab16c8. URL: https://dx.doi.org/10.1088/1742-5468/ab16c8.

[4] Auré lien Decelle, Cyril Furtlehner, and Beatriz Seoane. "Equilibrium and non-equilibrium regimes in the learning of restricted Boltzmann machines∗". In: *Journal of Statistical Mechanics: Theory and Experiment* 2022.11 (Nov. 2022), p. 114009. DOI: 10.1088/1742-5468/ac98a7. URL: https://doi.org/10.1088%2F1742-5468%2Fac98a7.

[5] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. "A Fast Learning Algorithm for Deep Belief Nets". In: *Neural Computation* 18.7 (July 2006), pp. 1527–1554. ISSN: 0899-7667. DOI: 10.1162/neco.2006.18.7.1527. eprint: https://direct.mit.edu/neco/article-pdf/18/7/1527/816558/neco.2006.18.7.1527.pdf. URL: https://doi.org/10.1162/neco.2006.18.7.1527.

[6] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *Nature* 521.7553 (May 2015), pp. 436–444. DOI: 10.1038/nature14539. URL: https://ideas.repec.org/a/nat/nature/v521y2015i7553d10.1038_nature14539.html.

[7] Honglak Lee et al. "Unsupervised Learning of Hierarchical Representations with Convolutional Deep Belief Networks". In: *Commun. ACM* 54.10 (Oct. 2011), pp. 95–103. ISSN: 0001-0782. DOI: 10.1145/2001269.2001295. URL: https://doi.org/10.1145/2001269.2001295.

[8] Juyong Song, Matteo Marsili, and Junghyo Jo. "Resolution and relevance trade-offs in deep learning". In: *Journal of Statistical Mechanics: Theory and Experiment* 2018.12 (Dec. 2018), p. 123406. DOI: 10.1088/1742-5468/aaf10f. URL: https://dx.doi.org/10.1088/1742-5468/aaf10f.

[9] Rongrong Xie and Matteo Marsili. *Occam learning*. 2022. arXiv: 2210.13179 [cond-mat.dis-nn].