Better Mixing via Deep Representations

Yoshua Bengio[1] CheckMy@WebPage.ca
Dept. IRO, Université de Montréal. Montréal (QC), H2C 3J7, Canada

Grégoire Mesnil[1] CheckMy@WebPage.ca
Dept. IRO, Université de Montréal. Montréal (QC), H2C 3J7, Canada
LITIS EA 4108, Université de Rouen. 768000 Saint Etienne du Rouvray, France

Yann Dauphin CheckMy@WebPage.ca
Salah Rifai CheckMy@WebPage.ca
Dept. IRO, Université de Montréal. Montréal (QC), H2C 3J7, Canada

Abstract

It has been hypothesized, and supported with experimental evidence, that deeper representations, when well trained, tend to do a better job at disentangling the underlying factors of variation. We study the following related conjecture: better representations, in the sense of better disentangling, can be exploited to produce Markov chains that mix faster between modes. Consequently, mixing between modes would be more efficient at higher levels of representation. To better understand this, we propose a secondary conjecture: the higher-level samples fill more uniformly the space they occupy and the high-density manifolds tend to unfold when represented at higher levels. The paper discusses these hypotheses and tests them experimentally through visualization and measurements of mixing between modes and interpolating between samples.

1. Introduction and Background

Deep learning algorithms attempt to discover multiple levels of representation of the given data (see (Bengio, 2009) for a review), with higher levels of representation defined hierarchically in terms of lower level ones. The central motivation is that higher-level representations can potentially capture relevant higher-level abstractions. Mathematical results in the case of specific function families have shown that choosing a sufficient depth of representation can yield exponential benefits, in terms of size of the model, to represent some functions (Håstad, 1986; Håstad and Goldmann, 1991; Bengio *et al.*, 2006; Bengio and LeCun, 2007; Bengio and Delalleau, 2011). The intuition behind these theoretical advantages is that lower-level features or latent variables can be *re-used* in many ways to construct higher-level ones, and the potential gain becomes exponential with respect to depth of the circuit that relates lower-level features and higher-level ones (thanks to the exponential number of paths in between). The ability of deep learning algorithms to construct abstract features or latent variables on top of the observed variables relies on this idea of re-use, which brings with it not only computational but also statistical advantages in terms of *sharing of statistical strength*, e.g., as already exploited in multi-task learning (Caruana, 1995; Baxter, 1997; Collobert and Weston, 2008) and learning algorithms involving *parameter sharing* (Lang and Hinton, 1988; LeCun, 1989).

There is another – less commonly discussed – motivation for deep representations, introduced in Bengio (2009): the idea that they may help to *disentangle* the underlying factors of variation. Clearly, if we had learning algorithms that could do a good job of discovering and separating out the underlying causes and factors of variation present in the data, it would make further processing (typically, taking decisions) much easier. One could even say that the ultimate goal of AI research is to build machines that can understand the world around us, i.e., disentangle the factors and causes it involves, so progress in that direction seems important. If learned representations do a good job of disentangling the underlying factors of variation, earning (on top of these representations, e.g., towards specific tasks of interest) becomes substantially easier

[1]Indicates equal contribution

Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28.

because disentangling counters the effects of the curse of dimensionality. With good disentangling, there is no need for further learning, only good inference. Several observations suggest that some deep learning algorithms indeed help to disentangle the underlying factors of variation (Goodfellow *et al.*, 2009; Glorot *et al.*, 2011). However, it is not clear why, and to what extent in general (if any), different deep learning algorithms may sometimes help this disentangling.

Many deep learning algorithms are based on some form of unsupervised learning, hence capturing salient structure in the data distribution. Whereas deep learning algorithms have mostly been used to learn features and exploit them for classification or regression tasks, their unsupervised nature also means that in several cases they can be used to generate samples. In general the associated sampling algorithms involve a Markov Chain and MCMC techniques, and these can notoriously suffer from a fundamental problem of *mixing between modes*: it is difficult for the Markov chain to jump from one mode of the distribution to another, when these are separated by large low-density regions, a common situation in real-world data, and under the *manifold hypothesis* (Cayton, 2005; Narayanan and Mitter, 2010). This hypothesis states that natural classes present in the data are associated with low-dimensional regions in input space (manifolds) near which the distribution concentrates, and that different class manifolds are well-separated by regions of very low density. Slow mixing between modes means that consecutive samples tend to be correlated (belong to the same mode) and that it takes many consecutive sampling steps to go from one mode to another and even more to cover all of them, i.e., to obtain a large enough representative set of samples (e.g. to compute an expected value under the target distribution). This happens because these jumps through the empty low-density void between modes are unlikely and rare events. When a learner has a poor model of the data, e.g., in the initial stages of learning, the model tends to correspond to a smoother and higher-entropy (closer to uniform) distribution, putting mass in larger volumes of input space, and in particular, between the modes (or manifolds). This can be visualized in generated samples of images, that look more blurred and noisy. Keep in mind that MCMCs tend to make moves to *nearby probable configurations*. Mixing between modes is therefore initially easy for such poor models. However, as the model improves and its corresponding distribution sharpens near where the data concentrate, mixing between modes becomes considerably slower. Making one unlikely move (i.e., to a low-probability configuration) may be possible, but making N such moves becomes exponentially unlikely in N, as illustrated in Figure 1. Since sampling is an integral part of many learning algorithms (e.g., to estimate the log-likelihood gradient), slower mixing between modes then means slower or poorer learning, and one may even suspect that learning stalls at some point because of the limitations of the sampling algorithm. To improve mixing between modes, a powerful idea that has been explored recently for deep learning algorithms (Desjardins *et al.*, 2010; Cho *et al.*, 2010; Salakhutdinov, 2010b;a) is *tempering* (Neal, 1994). The idea is to use smoother densities (associated with *higher temperature* in a Boltzmann Machine or Markov Random Field formulation) to make quick but approximate jumps between modes, but use the sharp "correct" model to actually generate the samples of interest around these modes, and allow samples to be exchanged between the different levels of temperature.

Here we want to discuss another possibly related idea, and claim that *mixing between modes is easier when sampling at the higher levels of representation*. The objective is not to propose a new sampling algorithm or a new learning algorithm, but rather to investigate this hypothesis through experiments using existing deep learning algorithms. The objective is to further our understanding of this hypothesis through more specific hypotheses aiming to explain why this would happen, using further experimental validation to test these more specific hypotheses. The idea that deeper generative models produce not only better features for classification but also better quality samples (in the sense of better corresponding to the target distribution being learned) is not novel and several observations support this hypothesis already, some quantitatively (Salakhutdinov and Hinton, 2009), some more qualitative (Hinton *et al.*, 2006). The specific contributions of this paper is to focus on why the samples may be better, and in particular, why the chains may converge faster, based on the previously introduced idea that deeper representations can do a better job of disentangling the underlying factors of representation.

2. Hypotheses

We first clarify the first hypothesis to be tested here.

> **Hypothesis H1: Depth vs Better Mixing Between Modes.** A successfully trained deeper architecture has the potential to yield representation spaces in which Markov chains mix faster between modes.

If experiments validate that hypothesis, the most important next question is: why? The main explanation we conjecture is formalized in the following hypothesis.

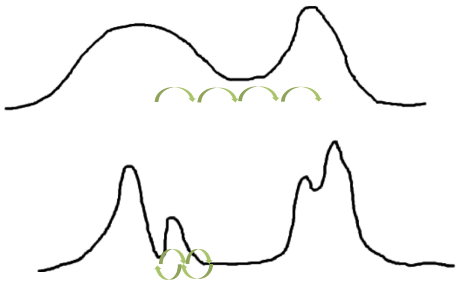


Figure 1. Top: early during training, MCMC mixes easily between modes because the estimated distribution has high entropy and puts enough mass everywhere for small-steps movements (MCMC) to go from mode to mode. Bottom: later on, training relying on good mixing can stall because estimated modes are separated by vast low-density deserts.

Hypothesis H2: Depth vs Disentangling. Part of the explanation of **H1** is that deeper representations can better disentangle the underlying factors of variation.

Why would that help to explain **H1**? Imagine an abstract (high-level) representation for object image data in which one of the factors is the "reverse video bit", which inverts black and white, e.g., flipping that bit replaces intensity $x \in [0, 1]$ by $1 - x$. With the default value of 0, the foreground object is dark and the background is light. Clearly, flipping that bit does not change most of the other semantic characteristics of the image, which could be represented in other high-level features. However, for every image-level mode, there would be a reverse-video counterpart mode in which that bit is flipped: these two modes would be separated by vast "empty" (low density) regions in input space, making it very unlikely for any Markov chain in input space (e.g. Gibbs sampling in an RBM) to jump from one of these two modes to another, because that would require most of the input pixels or hidden units of the RBM to simultaneously flip their value. Instead, if we consider the high-level representation which has a "reverse video" bit, flipping only that bit would be a very likely event under most Markov chain transition probabilities, since that flip would be a small change preserving high probability. As another example, imagine that some of the bits of the high-level representation identify the category of the object in the image, independently of pose, illumination, background, etc. Then simply flipping one of these object-class bits would also drastically change the raw pixel-space image, while keeping likelihood high. Jumping from an object-class mode to another would therefore be easy with a Markov chain in representation-space, whereas it would be much less likely in raw pixel-space.

Another point worth discussing (and which should be considered in future work) in **H2** is the notion of *degree* of disentangling. Although it is somewhat clear what a completely disentangled representation would look like, deep learning algorithms are unlikely to do a perfect job of disentangling, and current algorithms do it in stages, with more abstract features being extracted at higher levels. Better disentangling would mean that *some* of the learned features have a higher mutual information with *some* of the known factors. One would expect at the same time that the features that are highly predictive of one factor be less so of other factors, i.e., that they specialize to one or a few of the factors, becoming *invariant* to others. Please note here the difference between the objective of learning disentangled representations and the objective of learning invariant features (i.e., invariant to some specific factors of variation which are considered to be like nuisance parameters). In the latter case, one has to know ahead of time what the nuisance factors are (what is noise and what is signal?). In the former, it is not needed: we only seek to separate out the factors from each other. Some features should be sensitive to one factor and invariant to the others.

Let us now consider additional hypotheses that specialize **H2**.

Hypothesis H3: Disentangling Unfolds and Expands. Part of the explanation of **H2** is that more disentangled representations tend to **(a)** unfold the manifolds near which raw data concentrates, as well as **(b)** expand the relative volume occupied by high-probability points near these manifolds.

H3(a) says is that disentangling has the effect that the projection of high-density manifolds in the high-level representation space have a smoother density and are easier to model than the corresponding high-density manifolds in raw input space. Let us again use an object recognition analogy. If we have perfectly disentangled object identity, pose and illumination, the high-density manifold associated with the distribution of features in high-level representation-space is flat: we can interpolate between some training examples (i.e. likely configurations) and yet stay in a high-probability region. For example, we can imagine that interpolating between two images of the same object at different poses (lighting, position, etc.) in a high-level representation-space would yield images of the object at intermediate poses (i.e., corresponding to likely natural images), whereas interpolating in pixel space would give a superposition of the two original images (i.e., unlike any natural image). If interpolating between high-probability examples (i.e. within their con-

vex set) gives high-probability examples, then it means that the distribution is more uniform (fills the space) within that convex set, which is what **H3(b)** is saying. In addition, a good high-level representation does not need to allocate as much real estate (sets of values) for unlikely configurations. This is already what most unsupervised learning algorithms tend to do. For example, dimensionality reduction methods such as the PCA tend to define representations where most configurations are likely (but these only occupy a subspace of the possible raw-space configurations). Also, in clustering algorithms such as k-means, the training criterion is best minimized when clusters are approximately equally-weighted, i.e., the average posterior distribution over cluster identity is approximately uniform. Something similar is observed in the brain where different areas of somatosensory cortex correspond to different body parts, and the size of these areas adaptively depends (Flor, 2003) on usage of these (i.e., more frequent events are represented more finely and less frequent ones are represented more coarsely). Again, keep in mind that the actual representations learned by deep learning algorithms are not perfect, but what we will be looking for here is whether deeper representations correspond to more unfolded manifolds and to more locally uniform distributions, with high-probability points occupying an overall greater volume (compared to the available volume).

3. Representation-Learning Algorithms

The learning algorithms used in this paper to explore the preceding hypotheses are the Deep Belief Network or DBN (Hinton *et al.*, 2006), trained by stacking Restricted Boltzmann Machines or RBMs, and the Contractive Auto-Encoder or CAE (Rifai *et al.*, 2011a), for which a sampling algorithm was recently proposed (Rifai *et al.*, 2012). In the experiments, the distribution under consideration is the asymptotic distribution associated with the stochastic process used to generate samples. In the case of DBNs it clearly corresponds to the analytically defined distribution associated with the formula for the DBN probability. The Markov transition operator for DBNs is the one associated with Gibbs sampling (in the top RBM) (Hinton *et al.*, 2006). The Markov transition operator for stacked CAEs has been spelled out in Rifai *et al.* (2012) and linked to Langevin MCMC in Alain *et al.* (2012).

Each layer of the DBN is trained as an RBM, and a 1-layer DBN is just an RBM. An RBM defines a joint distribution between a hidden layer h and a visible layer v. Gibbs sampling at the top level of the DBN is used to obtain samples from the model: the sampled top-level representations are stochastically projected down to lower levels through the conditional distributions $P(v|h)$ defined in each RBM. To avoid unnecessary additional noise, and like previous authors have done, at the last stage of this process (i.e. to obtain the raw-input level samples), only the mean-field values of the visible units are used, i.e., $E[v|h]$. In the experiments on face data (where grey levels matter a lot), a Gaussian RBM is used at the lowest level.

An auto-encoder (LeCun, 1987; Hinton and Zemel, 1994) is parametrized through an encoder function f mapping input-space vector x to representation-space vector h, and a decoder function g mapping representation-space vector h to input-space reconstruction r. The experiments with the CAE are with $h = f(x) = \text{sigmoid}(Wx + b)$ and $r = g(h) = \text{sigmoid}(W^T h + c)$. The CAE is a regularized auto-encoder with tied weights (input to hidden weights are the transpose of hidden to reconstruction weights). Let $J = \frac{\partial f(x)}{\partial x}$ the Jacobian matrix of the encoder function. The CAE is trained to minimize a cross-entropy reconstruction loss plus a contractive regularization penalty $\alpha||J||_F^2$ (the sum of the squared elements of the Jacobian matrix). Like RBMs, CAE layers can be stacked to form deeper models, and one can either view them as deep auto-encoders (by composing the encoders together and the decoders together) or like in a DBN, as a top-level generative model (from which one can sample) coupled with encoding and decoding functions into and from the top level (by composing the lower-level encoders and decoders). A sampling algorithm was recently proposed for CAEs (Rifai *et al.*, 2012), alternating between projecting through the auto-encoder (i.e. performing a reconstruction) and adding Gaussian noise $JJ^T\epsilon$ in the directions of variation captured by the auto-encoder.

4. Experiments

The experiments have been performed on the MNIST digits dataset (LeCun *et al.*, 1998) and the Toronto Face Database (Susskind *et al.*, 2010), TFD. The former has been heavily used to evaluate many deep learning algorithms, while the latter is interesting because of the manifold structure it displays, and for which the main control factors (such as emotion and person identity) are known.

We have varied the number of hidden units at each layer independently for shallow and deep networks, and the models that gave best validation performance for each depth are shown. The qualitative aspects of the results were insensitive to layer size. The results reported are for DBNs with 768-1024-1024 layer sizes (28×28 input) on MNIST, and 2304-512-1024 on TFD

(48×48 input). The CAEs have sizes 768-1000-1000 and 2304-1000-1000 on MNIST and TFD respectively.

4.1. Sampling at Different Depths

4.1.1. Better Samples at Higher Levels

To test **H1**, we first plot sequences of samples at various depths. One can verify in Fig. 2 that samples obtained at deeper layers are visually more likely and mix faster between modes.

In addition, we measure the quality of the obtained samples, using a procedure for the comparison of sample generators described in Breuleux *et al.* (2011). Note that the mixing properties measured here are a consequence of the underlying models as well as of the chosen sampling procedures. For this reason, we have chosen to monitor the *quality of the samples* with respect to the original data generating distribution that was used to train the model. The procedure of Breuleux *et al.* (2011) measures the log-likelihood of a test set under the density computed from a Parzen window density estimator built on generated samples ($10,000$ samples here). Log-likelihoods for different models are presented in Table 1 (rightmost columns). Those results also suggest that the quality of the samples is higher if the Markov chain process used for sampling takes place in the upper layers.

This observation agrees with **H3(b)** that moving in higher-level representation spaces where the manifold has been expanded provides higher quality samples than moving in the raw input space where it may be hard to stay in high density regions.

4.1.2. Visualizing Representation-Space by Interpolating Between Neighbors

According to **H3(a)**, deeper layers tend to locally unfold the manifold near high-densities regions of the input space, while according to **H3(b)** there should be more relative volume taken by plausible configurations in representation-space. Both of these would imply that convex combinations of neighboring examples in representation-space correspond to more likely input configurations. Indeed, interpolating between points on a flat manifold should stay on the manifold. Furthermore, when interpolating between examples of different classes (i.e., different modes), **H3(b)** would suggest that most of the points in between (on the linear interpolation line) should correspond to plausible samples, which would not be the case in input space. In Fig. 3, we interpolate linearly between neighbors in representation space and visualize in the input space the interpolated points obtained at various depths. One can see that interpolating at deeper levels gives visually more plausible samples.

4.2. Measuring Mixing Between Modes by Counting Number of Classes Visited

We evaluate here the ability of mixing among various classes. We consider sequences of length 10, 20 or 100 and compute histograms of number of different classes visited in a sequence, for the two different depths and learners, on TFD. Since classes typically are in different modes (manifolds), counting how many different classes are visited in an MCMC run tells us how quickly the chain mixes between modes. We have chosen this particular method to monitor mixing modes because it focuses more directly on visits to modes, instead of the traditional autocorrelation of the samples (which measures how fast the samples change). Fig. 4(c,f) show that the deeper architectures visit more classes and the CAE mixes faster than the DBN.

4.3. Occupying More Volume Around Data Points

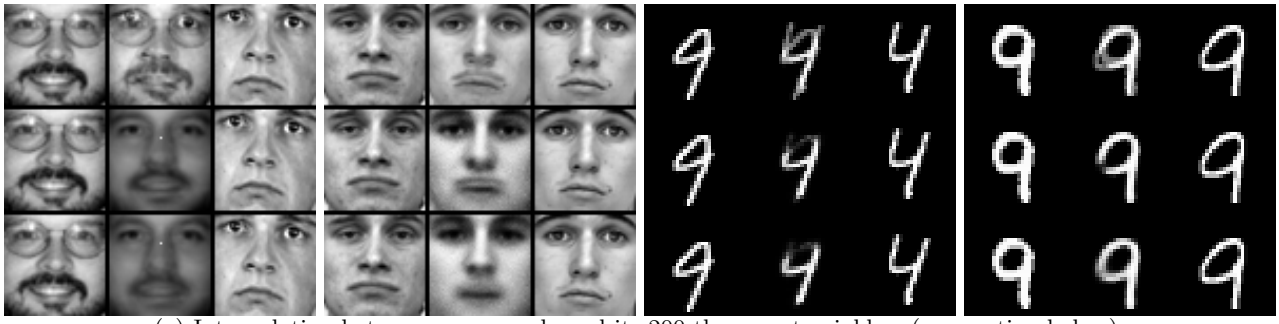
In these experiments (Fig. 4 (a,b,d,e)) we estimate the quality of samples whose representation is in the neighborhood of training examples, at different levels of representation. In the first setting (Fig. 4 (a,b)), the samples are interpolated at the midpoint between an example and its k-th nearest neighbor, with k on the x-axis. In the second case (Fig. 4 (d,e)), isotropic noise is added around an example, with noise standard deviation on the x-axis. In both cases, 500 samples are generated for each data point plotted on the figures, with the y-axis being the log-likelihood introduced earlier, i.e., estimating the quality of the samples. We find that on higher-level representations of both the CAE and DBN, a much larger proportion of the local volume is occupied by likely configurations, i.e., closer to the input-space manifold near which the actual data-generating distribution concentrates. Whereas the first experiment shows that this is true in the convex set between neighbors at different distances (i.e., in the directions of the manifold), the second shows that this is also true in random directions locally (but of course likelihoods are also worse there). The first result therefore agrees with **H3(a)** (unfolding) and **H3(b)** (volume expansion), while the second result mostly confirms **H3(b)**.

4.4. Discriminative Ability vs Volume Expansion

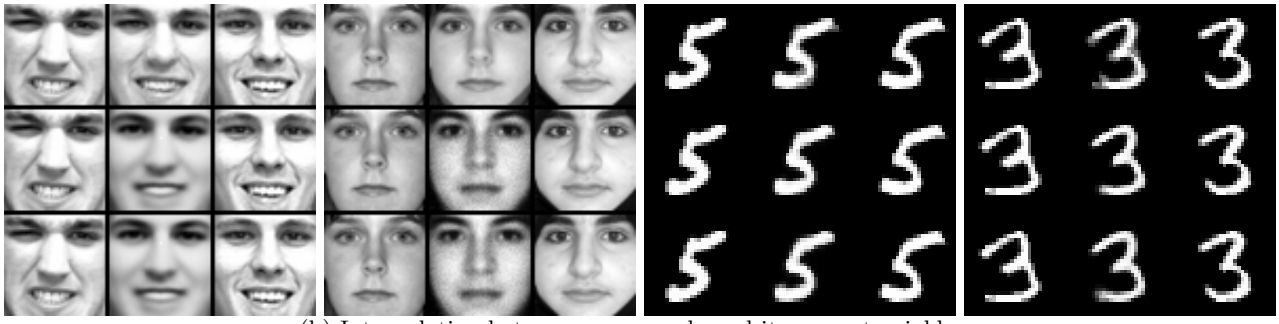
Hypothesis **H3** could arguably correspond to worse discriminative power[2]: if on the higher-level representations the different classes are "closer" to each other

[2] as pointed out by Aaron Courville, personal communication

Figure 2. Sequences of 25 samples generated with a CAE on TFD (rows 1 and 2, respectively for 1 or 2 hidden layers) and with an RBM on MNIST (rows 3 and 4, respectively for 1 or 2 hidden layers). On TFD, the second layer clearly allows to get quickly from woman samples (left) to man samples (right) passing by various facial expressions whereas the single hidden layer model shows poor samples. Bottom rows: On MNIST, the single-layer model gets stuck near the same mode while the second layer allows to mix among classes.



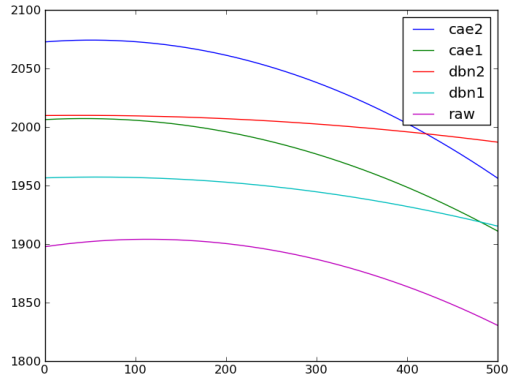
(a) Interpolating between an example and its 200-th nearest neighbor (see caption below).



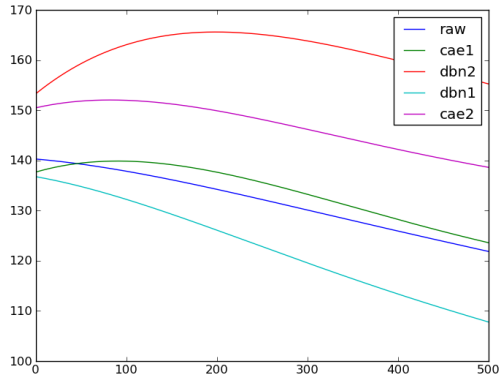
(b) Interpolating between an example and its nearest neighbor.

(c) Sequences of points interpolated at different depths

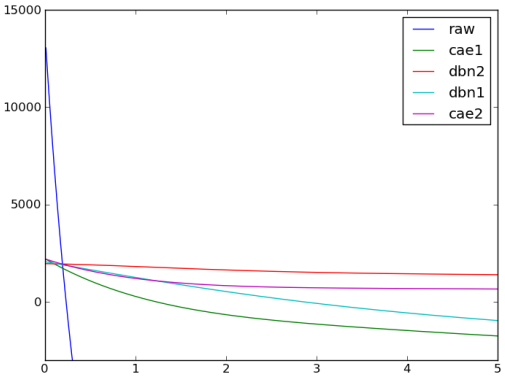
Figure 3. Linear interpolation between a data sample and the 200-th (a) and 1st (b) nearest neighbor, using representations at various depths (top row=input space, middle row=1st layer, bottom row=2nd layer). In each 3×3 block the left and right columns are test examples while the middle column is the image obtained by interpolation, based on different levels of representation. Interpolating at higher levels clearly gives more plausible samples. Especially in the raw input space (e.g., (a), 2nd block), one can see two mouths overlapping while only one mouth appears for the interpolated point at the 2nd layer. Interpolating with the 1-nearest neighbor does not show any difference between the levels because the nearest neighbors are close enough for a linear interpolation to be meaningful, while interpolaing with the 200-th nearest neighbors shows the failure of interpolation in raw input space but successful interpolation in deeper levels. In (c), we interpolate between samples of *different classes*, at different depths (top=raw input, middle=1st layer, bottom=2nd layer). Note how in lower levels one has to go through unplausible patterns, whereas in the deeper layers one almost jumps from a high-density region of one class to another (of the other class).



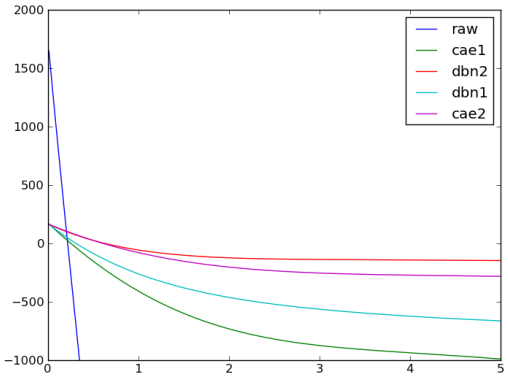
(a) TFD



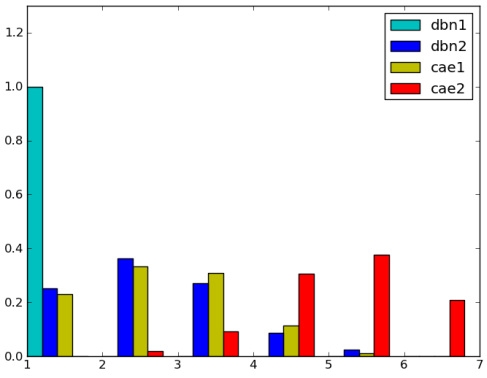
(b) MNIST



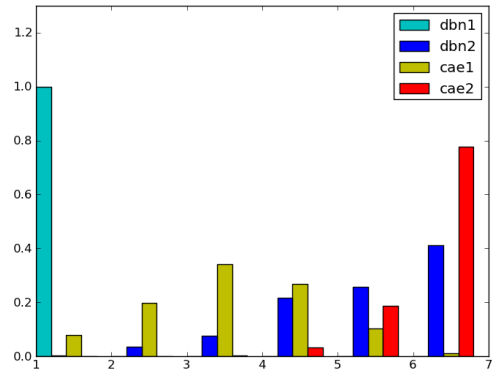
(c) TFD



(d) MNIST



(e) Mixing - 10 samples



(f) Mixing - 20/100 samples

Figure 4. (a) (b) Local Convex Hull - Log-density computed w.r.t. linearly interpolated samples between an example and its k-NNs, for k (x-axis) between 1 and 500. The manifold thus seem generally more unfolded (flatter) in deeper levels, especially against raw input space, as interpolating between far points yields higher density under deeper representations. (c) (d) Local Convex Ball - Log-density of samples generated by adding Gaussian noise to representation at different levels ($\sigma \in [0.01, 5]$, the x-axis): More volume is occupied by good samples on deeper layers. (e) (f) Mode Mixing Histograms - distribution (y-axis) of number of classes visited (x-axis) for different models. (e) with 10 samples. (f) with 20 samples for CAE, 100 samples with DBN. Deeper models mix much better.

	Classification				Log-likelihood	
	MNIST		TFD		MNIST	TFD
	SVM	MLP+	SVM	MLP+		
raw	8.34%	-	33.48 ±2.14 %	-	-	-
CAE-1	1.97%	1.14%	25.44 ±2.45%	24.12 ±1.87 %	67.69 ±2.87	591.90 ±12.27
CAE-2	1.73%	0.81%	24.76 ±2.46%	23.73 ±1.62%	121.17 ±1.59	2110.09 ±49.15
DBN-1	1.62%	1.21%	26.85 ±1.62%	28.14±1.40	−243.91 ±54.11	604 ±14.67
DBN-2	1.33%	0.99%	26.54 ±1.91%	27.79 ±2.34	137.89 ±2.11	1908.80 ±65.94

Table 1. Left: Classification rates of various classifiers (SVM, MLP+) using representations at various depth (with CAE or DBN) learned on the MNIST and TFD datasets. The DBN 0.99% error on MNIST has been obtained with a 3-layer DBN and the 0.81% error with the Manifold tangent Classifier (Rifai *et al.*, 2011b) that is based on a CAE-2 and discriminant fine-tuning. MLP+ uses discriminant fine-tuning. Right: Log-likelihoods from Parzen-Windows density estimators based on $10,000$ samples generated by each model. This quantitatively confirms that the samples generated from deeper levels are of higher quality, in the sense of better covering the zones where test examples are found.

(making it easier to mix between them), would it not mean that they are more confusable? We first confirm with the tested models (as a sanity check) that the deeper level features are conducive to better classification performance, in spite of their better generative abilities and better mixing between modes.

We train a linear SVM on the concatenation of the raw input with the upper layers representations (which worked better than using only the top layer, a setup already used successfully when there is no supervised fine-tuning (Lee *et al.*, 2009)). Results presented in Table 1 show that the representation is more linearly separable if one increases the depth of the architecture and the information added by each layer is helpful for classification. Also, fine-tuning a MLP initialized with those weights is still the best way to reach state-of-the-art performances.

To explain the good discriminant abilities of the deeper layers (either when concatenated with lower layers or when fine-tuned discriminatively) in spite of the better mixing observed, we conjecture the help of a better disentangling of the underlying factors of variation, and in particular of the class factors. This would mean that the manifolds associated with different classes are more unfolded (as assumed by **H3(a)**) and possibly that different hidden units specialize more to specific classes than they would on lower layers. Hence the unfolding (**H3(a)**) and disentangling (**H1**) hypotheses reconcile better discriminative ability with expanded volume of good samples (**H3(b)**).

5. Conclusion

The following hypotheses were tested: (1) deeper representations can yield better samples and better mixing between modes; (2) this is due to better disentangling; (3) this is associated with unfolding of the manifold where data concentrate along with expansion of the volume good samples take in representation-space. The experimental results were in agreement with these hypotheses. They showed better samples and better mixing on higher levels, better samples obtained when interpolating between examples at higher levels, and better samples obtained when adding isotropic noise at higher levels. We also considered the potential conflict between the third hypothesis and better discrimination (confirmed on the models tested) and explained it away as a consequence of the second hypothesis.

This could be immediate good news for applications requiring to generate MCMC samples: by transporting the problem to deeper representations, better and faster results could be obtained. Future work should also investigate the link between better mixing and the process of training deep learners themselves, when they depend on an MCMC to estimate the log-likelihood gradient. One interesting direction is to investigate the link between tempering and the better mixing chains obtained from deeper layers.

Acknowledgements

The authors thank A. Courville and P. Vincent for fruitful discussions, as well as NSERC, Canada Research Chairs, CIFAR, Compute Canada and by the French ANR Project ASAP ANR-09-EMER-001. Codes for the experiments have been implemented using the Theano (Bergstra *et al.*, 2010) Machine Learning libraries.

References

Alain, G., Bengio, Y., and Rifai, S. (2012). Regularized auto-encoders estimate local statistics. Technical Report Arxiv report 1211.4246, Université de Montréal.

Baxter, J. (1997). A Bayesian/information theoretic model of learning via multiple task sampling. *Machine Learning*, **28**, 7–40.

Bengio, Y. (2009). Learning deep architectures for AI.

Foundations and Trends in Machine Learning, **2**(1), 1–127. Also published as a book. Now Publishers, 2009.

Bengio, Y. and Delalleau, O. (2011). On the expressive power of deep architectures. In *ALT'2011*.

Bengio, Y. and LeCun, Y. (2007). Scaling learning algorithms towards AI. In L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, editors, *Large Scale Kernel Machines*. MIT Press.

Bengio, Y., Delalleau, O., and Le Roux, N. (2006). The curse of highly variable functions for local kernel machines. In *NIPS'05*, pages 107–114. MIT Press, Cambridge, MA.

Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D., and Bengio, Y. (2010). Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*.

Breuleux, O., Bengio, Y., and Vincent, P. (2011). Quickly generating representative samples from an rbm-derived process. *Neural Computation*, **23**(8), 2053–2073.

Caruana, R. (1995). Learning many related tasks at the same time with backpropagation. In *NIPS'94*, pages 657–664, Cambridge, MA. MIT Press.

Cayton, L. (2005). Algorithms for manifold learning. Technical Report CS2008-0923, UCSD.

Cho, K., Raiko, T., and Ilin, A. (2010). Parallel tempering is efficient for learning restricted Boltzmann machines. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN 2010)*, Barcelona, Spain.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In W. W. Cohen, A. McCallum, and S. T. Roweis, editors, *ICML 2008*, pages 160–167. ACM.

Desjardins, G., Courville, A., Bengio, Y., Vincent, P., and Delalleau, O. (2010). Tempered Markov chain monte carlo for training of restricted Boltzmann machine. In *JMLR W&CP: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2010)*, volume 9, pages 145–152.

Flor, H. (2003). Remapping somatosensory cortex after injury. *Advances in Neurology*, **83**, 195–204.

Glorot, X., Bordes, A., and Bengio, Y. (2011). Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the Twenty-eight International Conference on Machine Learning (ICML'11)*, volume 27, pages 97–110.

Goodfellow, I., Le, Q., Saxe, A., and Ng, A. (2009). Measuring invariances in deep networks. In *NIPS'2009*, pages 646–654.

Håstad, J. (1986). Almost optimal lower bounds for small depth circuits. In *Proceedings of the 18th annual ACM Symposium on Theory of Computing*, pages 6–20, Berkeley, California. ACM Press.

Håstad, J. and Goldmann, M. (1991). On the power of small-depth threshold circuits. *Computational Complexity*, **1**, 113–129.

Hinton, G. E. and Zemel, R. S. (1994). Autoencoders, minimum description length, and helmholtz free energy. In *NIPS'93*, pages 3–10. Morgan Kaufmann Publishers, Inc.

Hinton, G. E., Osindero, S., and Teh, Y. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, **18**, 1527–1554.

Lang, K. J. and Hinton, G. E. (1988). The development of the time-delay neural network architecture for speech recognition. Technical Report CMU-CS-88-152, Carnegie-Mellon University.

LeCun, Y. (1987). *Modèles connexionistes de l'apprentissage*. Ph.D. thesis, Université de Paris VI.

LeCun, Y. (1989). Generalization and network design strategies. Technical Report CRG-TR-89-4, University of Toronto.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, **86**(11), 2278–2324.

Lee, H., Pham, P., Largman, Y., and Ng, A. (2009). Unsupervised feature learning for audio classification using convolutional deep belief networks. In *NIPS'2009*.

Narayanan, H. and Mitter, S. (2010). Sample complexity of testing the manifold hypothesis. In *NIPS'2010*.

Neal, R. M. (1994). Sampling from multimodal distributions using tempered transitions. Technical Report 9421, Dept. of Statistics, University of Toronto.

Rifai, S., Vincent, P., Muller, X., Glorot, X., and Bengio, Y. (2011a). Contracting auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the Twenty-eight International Conference on Machine Learning (ICML'11)*.

Rifai, S., Dauphin, Y., Vincent, P., Bengio, Y., and Muller, X. (2011b). The manifold tangent classifier. In *NIPS'2011*. Student paper award.

Rifai, S., Bengio, Y., Dauphin, Y., and Vincent, P. (2012). A generative process for sampling contractive auto-encoders. In *ICML'2012*.

Salakhutdinov, R. (2010a). Learning deep Boltzmann machines using adaptive MCMC. In L. Bottou and M. Littman, editors, *Proceedings of the Twenty-seventh International Conference on Machine Learning (ICML-10)*, volume 1, pages 943–950. ACM.

Salakhutdinov, R. (2010b). Learning in Markov random fields using tempered transitions. In *NIPS'09*.

Salakhutdinov, R. and Hinton, G. E. (2009). Deep Boltzmann machines. In *AISTATS'2009*, volume 5, pages 448–455.

Susskind, J., Anderson, A., and Hinton, G. E. (2010). The Toronto face dataset. Technical Report UTML TR 2010-001, U. Toronto.