Learning Ordered Representations with Nested Dropout

Oren Rippel RIPPEL@MATH.MIT.EDU
Department of Mathematics, MIT; School of Engineering and Applied Sciences, Harvard University

Michael A. Gelbart MGELBART@SEAS.HARVARD.EDU
Program in Biophysics and School of Engineering and Applied Sciences, Harvard University

Ryan P. Adams RPA@SEAS.HARVARD.EDU
School of Engineering and Applied Sciences, Harvard University

Abstract

In this paper, we present results on ordered representations of data in which different dimensions have different degrees of importance. To learn these representations we introduce *nested dropout*, a procedure for stochastically removing coherent nested sets of hidden units in a neural network. We first present a sequence of theoretical results for the special case of a semi-linear autoencoder. We rigorously show that the application of nested dropout enforces identifiability of the units, which leads to an exact equivalence with PCA. We then extend the algorithm to deep models and demonstrate the relevance of ordered representations to a number of applications. Specifically, we use the ordered property of the learned codes to construct hash-based data structures that permit very fast retrieval, achieving retrieval in time logarithmic in the database size and independent of the dimensionality of the representation. This allows codes that are hundreds of times longer than currently feasible for retrieval. We therefore avoid the diminished quality associated with short codes, while still performing retrieval that is competitive in speed with existing methods. We also show that ordered representations are a promising way to learn adaptive compression for efficient online data reconstruction.

1. Introduction

The automatic discovery of representations is an increasingly important aspect of machine learning, motivated by a variety of considerations. For example, feature extraction is often a critical first step for supervised learning procedures. Representation learning enables one to avoid explicit feature engineering; indeed, approaches to deep feature learning have often found representations that outperform their hand-crafted counterparts (e.g., LeCun & Bengio, 1995; Hinton & Salakhutdinov, 2006; Vincent et al., 2010; Coates et al., 2011). In other situations, unsupervised representation learning is useful for finding low-dimensional manifolds for visualization (e.g., Tenenbaum et al., 2000; Roweis & Saul, 2000; Van der Maaten & Hinton, 2008). There has also been increasing interest in exploiting such representations for information retrieval, leveraging the ability of unsupervised learning to discover compact and concise codes that can be used to build efficient data structures (e.g., Weiss et al., 2008; Salakhutdinov & Hinton, 2009; Krizhevsky & Hinton, 2011).

One frustration associated with current representation learning techniques is redundancy from non-identifiability in the resulting encoder/decoder. That is, under standard models such as autoencoders, restricted Boltzmann machines, and sparse coding, any given solution is part of an *equivalence class* of solutions that are equally optimal. This class emerges from the invariance of the models to various transformations of the parameters. Permutation is one clear example of such a transformation, leading to a combinatorial number of equivalent representations for a given dataset and architecture. There exist many other types of redundancies as well; the optimality of an autoencoder solution is preserved under any invertible linear transformation of the innermost set of weights (Bourlard & Kamp, 1987). This degeneracy also poses a difficulty when comparing experiments, due to the lack of repeatability: a solution attained by the optimization procedure is extremely sensitive to the choice of initialization.

This large number of equivalent representations has an advantage, however: it provides flexibility in architecture de-

Proceedings of the 31^{st} International Conference on Machine Learning, Beijing, China, 2014. JMLR: W&CP volume 32.

sign. This freedom allows us to impose desirable structural constraints on the learned representations, without compromising their expressiveness. These constraints can imbue a number of useful properties, including the elimination of permutation non-identifiability. In this work we propose one such structural constraint: we specify *a priori* the quantity of information encapsulated in each dimension of the representation. This choice allows us to order the representation dimensions according to their information content.

The intuition behind our proposed approach to learning ordered representations is to train models such that the information contained in each dimension of the representation decreases as a function of the dimension index, following a pre-specified decay function. To this end, we introduce the *nested dropout* algorithm. As with the original dropout formulation (Hinton et al., 2012), nested dropout applies a stochastic mask over models. However, instead of imposing an independent distribution over each individual unit in a model, it assigns a distribution over nested subsets of representation units. More specifically, given a representation space of dimension K, we define a distribution $p_B(\cdot)$ over the representation index subsets $S_b = \{1, \ldots, b\}$, $b = 1, \ldots, K$. This has the property that if the j-th unit appears in a particular mask, then so do all "earlier" units $1, \ldots, j-1$, allowing the j-th unit to depend on them. This nesting leads to an inherent ordering over the representation dimensions. The distribution $p_B(\cdot)$ then governs the information capacity decay by modulating the relative frequencies of these masks. We motivate such ordered representations in several ways, described below.

Identifiability As discussed above, many current representation learning techniques suffer from non-identifiability of the solutions. We can remedy this by introducing strict representation ordering, which enforces distinguishability. We rigorously demonstrate this for the special case of a semi-linear autoencoder. We prove that the application of nested dropout leads to a significant reduction in the solution space complexity without harming the solution quality. Under an additional weak constraint, we further prove that the model has a single and unique global optimum. We show that this solution is exactly the set of eigenvalues of the covariance matrix of the data, ordered by eigenvalue magnitude. This demonstrates exact equivalence between semi-linear nested dropout autoencoders and principal component analysis (PCA).

Fast retrieval Current information retrieval procedures suffer from an intrinsic tradeoff between search speed and quality: representation dimensionality and dataset size must be sacrificed to gain search tractability (Grauman & Fergus (2013) offers an excellent overview of modern retrieval procedures). Given a query datum, a naïve brute force retrieval based on Hamming distance requires a linear scan of the database, which has complexity $\mathcal{O}(KN)$ where K is the code length and N the database size. Semantic hashing (Salakhutdinov & Hinton, 2009) retrieves examples within a Hamming neighborhood of radius R by directly scanning through all memory locations associated with them. This results in retrieval time complexity $\mathcal{O}(\binom{K}{R})$. While this is independent of the database size, it grows rapidly in K and therefore is computationally prohibitive even for codes tens of bits long; code length of 50 bits, for example, requires a petabyte of memory be addressed. Moreover, as the code length increases, it becomes very likely that many queries will not find any neighbors for any feasible radii. Locality sensitive hashing (Datar et al., 2004) seeks to preserve distance information by means of random projections; however, this can lead to very inefficient codes for high input dimensionality.

By imposing an ordering on the information represented in a deep model, we can learn hash functions that permit efficient retrieval. Because the importance of each successive coding dimension decays as we move through the ordering, we can naturally construct a binary tree data structure on the representation to capture a coarse-to-fine notion of similarity. This allows retrieval in time that is logarithmic with the dataset size and *independent* of the representation space dimensionality: the retrieval procedure adaptively selects the minimum number of code bits required for resolution. This enables very fast retrieval on large databases without sacrificing representation quality: we are able to consider codes hundreds of times longer than currently feasible with existing retrieval methods. For example, we perform retrieval on a dataset of a million entries of code length 2048 in an average time of 200μs per query—about 4 orders of magnitude faster than a linear scan or semantic hashing.

Adaptive compression Ordered representations can also be used for "continuous-degradation" lossy compression systems: they give rise to a continuous range of bitrate/quality combinations, where each additional bit corresponds to a small incremental increase in quality. This property can in principle be applied to problems such as video streaming. The representation only needs to be encoded a single time; then, users of different bandwidths can be adaptively sent codes of different length that exactly match their bitrates. The inputs can then be reconstructed optimally for the users' channel capacities.

2. Ordering with nested dropout

Dropout (Hinton et al., 2012) is a regularization technique for neural networks that adds stochasticity to the architecture during training. At each iteration, unbiased coins are flipped independently for each unit in the network, determining whether it is "dropped" or not. Every dropped unit is deleted from the network for that iteration, and an optimization step is taken with respect to the resulting network.

Nested dropout diverges from this in two main ways. First, only representation units are dropped. Second, instead of flipping independent coins for different units, we instead assign a prior distribution $p_B(\cdot)$ over the representation *indices* $1, \ldots, K$. We then sample an index $b \sim p_B(\cdot)$ and drop units $b+1, \ldots, K$. The sampled units then form nested subsets: if unit j appears in a network sample, then so do units $1, \ldots, j-1$. This nesting results in an inherent importance ranking of the representation dimensions, as a particular unit can always rely on the presence of its predecessors. For $p_B(\cdot)$ we select a geometric distribution: $p_B(b) = \rho^{b-1}(1-\rho)$. We make this choice due to the exponential decay of this distribution and its memoryless property (see Section 4.3).

Our architecture resembles an autoencoder in its parametric composition of an encoder and a decoder. We are given a set of N training examples $\{\mathbf{y}_n\}_{n=1}^N$ lying in space $\mathcal{Y} \subseteq \mathbb{R}^D$. We then transform the data into the *representation space* $\mathcal{X} \subseteq \mathbb{R}^K$ via a parametric transformation $\boldsymbol{f}_{\boldsymbol{\Theta}} : \mathcal{Y} \to \mathcal{X}$. We denote this function as the *encoder*, and label the representations as $\{\mathbf{x}_n\}_{n=1}^N \subset \mathcal{X}$. The *decoder* map $\boldsymbol{g}_{\boldsymbol{\Psi}} : \mathcal{X} \to \mathcal{Y}$ then reconstructs the inputs from their representations as $\{\hat{\mathbf{y}}_n\}_{n=1}^N$.

A single nested dropout sample Let us assume that we sample some $b \sim p_B(\cdot)$ and drop the last $K-b$ representation units; we refer to this case as the *b-truncation*. This structure is equivalent to an autoencoder with a representation layer of dimension b. For a given representation $\mathbf{x} \in \mathbb{R}^K$, we define $\mathbf{x}_{\downarrow b}$ as the truncation of the vector $\mathbf{x}$ where the last $K-b$ elements are removed.

Denoting the reconstruction of the b-truncation as $\hat{\mathbf{y}}_{\downarrow b} = \boldsymbol{g}_{\boldsymbol{\Psi}}(\boldsymbol{f}_{\boldsymbol{\Theta}}(\mathbf{y})_{\downarrow b})$, the reconstruction cost function associated with a b-truncation is then

$$C_{\downarrow b}(\boldsymbol{\Theta}, \boldsymbol{\Psi}) = \frac{1}{N} \sum_{n=1}^{N} \mathcal{L}\left(\mathbf{y}_n, \hat{\mathbf{y}}_{n\downarrow b}\right) \ . \tag{1}$$

In this work, we take the reconstruction loss $\mathcal{L}(\cdot, \cdot)$ to be the L_2 norm. Although we write this cost as a function of the full parametrization $(\boldsymbol{\Theta}, \boldsymbol{\Psi})$, due to the truncation only a subset of the parameters will contribute to the objective.

The nested dropout problem Given our distribution $p_B(\cdot)$, we consider the mixture of the different b-truncation objectives:

$$C(\boldsymbol{\Theta}, \boldsymbol{\Psi}) = \mathbb{E}_B\left[C_{\downarrow b}(\boldsymbol{\Theta}, \boldsymbol{\Psi})\right] = \sum_{b=1}^{K} p_B(b) C_{\downarrow b}(\boldsymbol{\Theta}, \boldsymbol{\Psi}). \tag{2}$$

We formulate the *nested dropout* problem as the optimization of this mixture with respect to the model parameters:

$$(\boldsymbol{\Theta}^*, \boldsymbol{\Psi}^*) = \arg\min_{\boldsymbol{\Theta}, \boldsymbol{\Psi}} \ C(\boldsymbol{\Theta}, \boldsymbol{\Psi}) \ . \tag{3}$$

2.1. Interpretation

Nested dropout has a natural interpretation in terms of information content in representation units. It was shown by Vincent et al. (2010) that training an autoencoder corresponds to maximizing a lower bound on the mutual information $\mathcal{I}(\boldsymbol{y}; \boldsymbol{x})$ between the input data and their representations. Specifically, the objective of the b-truncation problem can be written in the form

$$C_{\downarrow b}(\boldsymbol{\Theta}, \boldsymbol{\Psi}) \approx \mathbb{E}_{\mathbf{y}}\left[-\log p_{\boldsymbol{Y}|\boldsymbol{X}_{\downarrow b}}\left(\mathbf{y} \,|\, \boldsymbol{f}_{\boldsymbol{\Theta}}(\mathbf{y})_{\downarrow b}\,;\, \boldsymbol{\Psi}\right)\right] \tag{4}$$

where we assume our data are sampled from the true distribution $p_{\boldsymbol{Y}}(\cdot)$. The choice $p_{\boldsymbol{Y}|\boldsymbol{X}}\left(\mathbf{y} \,|\, \mathbf{x}\,;\, \boldsymbol{\Psi}\right) = \mathcal{N}(\mathbf{y}\,;\, \boldsymbol{g}_{\boldsymbol{\Psi}}(\mathbf{x}), \sigma^2 \mathbb{I}_D)$, for example, leads to the familiar autoencoder L_2 reconstruction penalty.

Now, define $\tilde{\mathcal{I}}_b(\boldsymbol{y}; \boldsymbol{x}) := -C_{\downarrow b}(\boldsymbol{\Theta}, \boldsymbol{\Psi}) \leq \mathcal{I}(\boldsymbol{y}; \boldsymbol{x})$ as the approximation of the true mutual information which we maximize for a given b. Then we can write the (negative) nested dropout problem in the form of a telescopic sum:

$$\begin{aligned} -C(\boldsymbol{\Theta}, \boldsymbol{\Psi}) &= \sum_{b=1}^{K} p_B(b) \tilde{\mathcal{I}}_b(\boldsymbol{y}; \boldsymbol{x}) \\ &= \tilde{\mathcal{I}}_1(\boldsymbol{y}; \boldsymbol{x}) + \sum_{b=2}^{K} \left[F_B(K) - F_B(b-1)\right] \Delta_b \ , \end{aligned} \tag{5}$$

where $F_B(b) = \sum_{b'=1}^{b} p_B(b')$ is the cumulative distribution function of $p_B(\cdot)$, and $\Delta_b := \tilde{\mathcal{I}}_b(\boldsymbol{y}; \boldsymbol{x}) - \tilde{\mathcal{I}}_{b-1}(\boldsymbol{y}; \boldsymbol{x})$ is the marginal information gained from increasing the representation dimensionality from b units to $b+1$.

This formulation provides a connection between the nested dropout objective and the optimal distribution of information across the representation dimensions. Note that the coefficients $F_B(K) - F_B(b)$ of the marginal mutual information are positive and monotonically decrease as a function of b regardless of the choice of distribution $p_B(\cdot)$. This establishes the ordering property intuitively sought by the nested dropout idea. We also see that if for some b we have $p_B(b) = 0$, i.e., index b has no support under $p_B(\cdot)$, then the ordering of representation dimensions b and $b-1$ no longer matters. If we set $p_B(1) = 0, \ldots, p_B(K-1) = 0$ and $p_B(K) = 1$, we recover the original order-free autoencoder formulation for K latent dimensions. In order to achieve strict ordering, then, the only assumption we must make is that $p_B(\cdot)$ has support over all representation indices. Indeed, this will be a sufficient condition for our proofs in Section 3.2. Equation (5) informs us of how our prior choice of $p_B(\cdot)$ dictates the optimal information allocation per unit.

3. Exact recovery of PCA

In this section, we apply nested dropout to a semi-linear autoencoder. This model has a linear or a sigmoidal encoder, and a linear decoder. The relative simplicity of this case

allows us to rigorously study the ordering property implied by nested dropout.

First, we show that the class of optimal solutions of the nested dropout autoencoder is a subset of the class of optimal solutions of a standard autoencoder. This means that introducing nested dropout does not sacrifice the quality of the autoencoder solution. Second, we show that equipping an autoencoder with nested dropout significantly constrains its class of optimal solutions. We characterize these restrictions. Last, we show that under an additional orthonormality constraint, the model features a single, unique solution that is exactly the set of K eigenvectors with the largest magnitudes arising from the covariance matrix of the inputs, ordered by decreasing eigenvalue magnitude. Hence this recovers the PCA solution exactly. This is in contrast to a standard autoencoder, which recovers the PCA solution up to an invertible linear map.

3.1. Problem definitions and prior results

The standard linear autoencoder problem Given our inputs, we apply the linear encoder $\boldsymbol{f}_{\boldsymbol{\Theta}}(\mathbf{y}) := \boldsymbol{\Omega}\mathbf{y} + \boldsymbol{\omega}$ with parameters $\boldsymbol{\Omega} \in \mathbb{R}^{K\times D}$ and bias vector $\boldsymbol{\omega} \in \mathbb{R}^K$ for $K \leq D$. Our proofs further generalize to sigmoidal nonlinearities applied to the output of the encoder, but we omit these for clarity. The decoder map $\boldsymbol{g}_{\boldsymbol{\Psi}} : \mathcal{X} \to \mathcal{Y}$ is similarly taken to be $\boldsymbol{g}_{\boldsymbol{\Psi}}(\mathbf{x}) := \boldsymbol{\Gamma}\mathbf{x} + \boldsymbol{\gamma}$ with parameters $\boldsymbol{\Gamma} \in \mathbb{R}^{D\times K}$ and $\boldsymbol{\gamma} \in \mathbb{R}^D$. We also define the design matrices $\boldsymbol{Y}$ and $\boldsymbol{X}$ whose columns consist of the observations and their representations, respectively.

The reconstruction of each datum is then defined as the composition of the encoder and decoder maps. Namely, $\hat{\mathbf{y}}_n = \boldsymbol{\Gamma}(\boldsymbol{\Omega}\mathbf{y}_n + \boldsymbol{\omega}) + \boldsymbol{\gamma}\ \forall n = 1, \ldots, N$. A semi-linear autoencoder seeks to minimize the reconstruction cost

$$C(\boldsymbol{\Theta}, \boldsymbol{\Psi}) = \sum_{n=1}^{N} \left\|\mathbf{y}_n - \boldsymbol{g}_{\boldsymbol{\Psi}}(\boldsymbol{f}_{\boldsymbol{\Theta}}(\mathbf{y}_n))\right\|^2 \quad (6)$$

$$= \left\|\boldsymbol{Y} - (\boldsymbol{\Gamma}(\boldsymbol{\Omega}\boldsymbol{Y} + \boldsymbol{\omega}) + \boldsymbol{\gamma})\right\|_F^2 \quad (7)$$

where by $\|\cdot\|_F$ we denote the Frobenius matrix norm. From this point on, without loss of generality we assume that $\boldsymbol{\omega} = \mathbf{0}$, $\boldsymbol{\gamma} = \mathbf{0}$, and that the data is zero-centered. All our results hold otherwise, but with added shifting constants.

A single b-truncation problem We continue to consider the b-truncation problem, where the last $K - b$ units of the representation are dropped. As before, for a given representation $\mathbf{x} \in \mathbb{R}^K$, we define $\mathbf{x}_{\downarrow b}$ to be the truncation of vector $\mathbf{x}$. Defining the truncation matrix $\boldsymbol{J}_{m\to n} \in \mathbb{R}^{n\times m}$ as $[\boldsymbol{J}_{m\to n}]_{ab} = \delta_{ab}$, then $\mathbf{x}_{\downarrow b} = \boldsymbol{J}_{K\to b}\mathbf{x}$. The decoder is then written as $\boldsymbol{g}_{\boldsymbol{\Psi}\downarrow b}(\mathbf{x}_{\downarrow b}) = \boldsymbol{\Gamma}_{\downarrow b}\mathbf{x}_{\downarrow b}$, where we write $\boldsymbol{\Gamma}_{\downarrow b} = \boldsymbol{\Gamma}\boldsymbol{J}_{K\to b}^T$ in which the last $K - b$ columns of $\boldsymbol{\Gamma}$ are removed. The reconstruction cost function associated with a b-truncation is then

$$C_{\downarrow b}(\boldsymbol{\Theta}_{\downarrow b}, \boldsymbol{\Psi}_{\downarrow b}) = \left\|\boldsymbol{Y} - \Gamma_{\downarrow b}\boldsymbol{X}_{\downarrow b}\right\|_F^2 . \quad (8)$$

We define $(\boldsymbol{\Theta}^*_{\downarrow b}, \boldsymbol{\Psi}^*_{\downarrow b}) = \arg\min_{\boldsymbol{\Theta}_{\downarrow b}, \boldsymbol{\Psi}_{\downarrow b}} \boldsymbol{C}_{\downarrow b}(\boldsymbol{\Theta}_{\downarrow b}, \boldsymbol{\Psi}_{\downarrow b})$ to be an optimal solution of the b-truncation problem; we label the corresponding optimal cost as $C^*_{\downarrow b}$. Also, let $\boldsymbol{V}_{\boldsymbol{Y}} = \boldsymbol{Y}\boldsymbol{Y}^T$ be (proportional to) the empirical covariance matrix of $\{\mathbf{y}_n\}_{n=1}^N$ with eigendecomposition $\boldsymbol{V}_{\boldsymbol{Y}} = \boldsymbol{Q}\boldsymbol{\Sigma}^2\boldsymbol{Q}^T$, where $\boldsymbol{\Sigma}^2$ is the diagonal matrix constituting of the eigenvalues arranged in decreasing magnitude order, and $\boldsymbol{Q}$ the orthonormal matrix of the respective eigenvectors. Similarly, let $\boldsymbol{R}$ be the orthonormal eigenvector matrix of $\boldsymbol{Y}^T\boldsymbol{Y}$, arranged by decreasing order of eigenvalue magnitude.

The b-truncation problem exactly corresponds to the original semi-linear autoencoder problem, where the representation dimension is taken to be b in the first place. As such, we can apply known results about the form of the solution of a standard autoencoder. It was proven in Bourlard & Kamp (1987) that this optimal solution must be of the form

$$\boldsymbol{X}^*_b = \boldsymbol{T}_b\boldsymbol{\Sigma}_{\downarrow b}\boldsymbol{R}^T \qquad \boldsymbol{\Gamma}^*_b = \boldsymbol{Q}_{\downarrow b}\boldsymbol{T}_b^{-1} \quad (9)$$

where $\boldsymbol{T}_b \in \mathbb{R}^{b\times b}$ is an invertible matrix, $\boldsymbol{\Sigma}_{\downarrow b} = \boldsymbol{J}_{K\to b}\boldsymbol{\Sigma} \in \mathbb{R}^{b\times D}$ the matrix with the b largest-magnitude eigenvalues, and $\boldsymbol{Q}_{\downarrow b} = \boldsymbol{Q}\boldsymbol{J}_{K\to b}^T \in \mathbb{R}^{D\times b}$ the matrix with the b corresponding eigenvectors. This result was established for an autoencoder of representation dimension b; we reformulated the notation to suit the nested dropout problem we define in the next subsection.

It can be observed from Equation (9) that the semi-linear autoencoder has a strong connection to PCA. An autoencoder discovers the eigenvectors of the empirical covariance matrix of $\{\mathbf{y}_n\}_{n=1}^N$ corresponding to its b eigenvalues of greatest magnitude; however, this is up to to an invertible linear transformation. This class includes rotations, scalings, reflections, index permutations, and so on. This non-identifiability has an undesirable consequence: it begets a huge class of optimal solutions.

The nested dropout problem We now introduce the nested dropout problem. Here, we assign the distribution $b \sim p_B(\cdot)$ as a prior over b-truncations. For our proofs to hold our only assumption about this distribution is that it has support over the entire index set, i.e., $p_B(b) > 0,\ \forall b = 1, \ldots, K$. To that end, we seek to minimize the nested dropout cost function, which we define as the mixture of the K truncated models under $p_B(\cdot)$:

$$C(\boldsymbol{\Theta}, \boldsymbol{\Psi}) = \mathbb{E}_B\left[\left\|\boldsymbol{Y} - \Gamma_{\downarrow b}\boldsymbol{X}_{\downarrow b}\right\|_F^2\right] \quad (10)$$

$$= \sum_{b=1}^{K} p_B(b)\left\|\boldsymbol{Y} - \Gamma_{\downarrow b}\boldsymbol{X}_{\downarrow b}\right\|_F^2 . \quad (11)$$

3.2. The nested dropout problem recovers PCA exactly

Below we provide theoretical justification for the claims made in the beginning of this section. All of the proofs can be found in the supplementary material.

Theorem 1. *Every optimal solution of the nested dropout problem is necessarily an optimal solution of the standard autoencoder problem.*

Definition. *We define matrix* $\boldsymbol{T} \in \mathbb{R}^{K \times K}$ *to be* commutative in its truncation and inversion *if each of its leading principal minors* $\boldsymbol{J}_{K \to b} \boldsymbol{T} \boldsymbol{J}_{K \to b}^T$, $b = 1, \ldots, K$ *is invertible, and the inverse of each of its leading principal minors is equal to the leading principal minor of the inverse* $\boldsymbol{T}^{-1}$*, namely*

$$\boldsymbol{J}_{K \to b} \boldsymbol{T}^{-1} \boldsymbol{J}_{K \to b}^T = (\boldsymbol{J}_{K \to b} \boldsymbol{T} \boldsymbol{J}_{K \to b}^T)^{-1} \,. \tag{12}$$

The below theorem, combined with Lemma 1, establishes tight constraints on the class of optimal solutions of the nested dropout problem. For example, an immediate corollary of this is that $\boldsymbol{T}$ cannot be a permutation matrix, as for such a matrix there must exist some leading principal minor that is not invertible.

Theorem 2. *Every optimal solution of the nested dropout problem must be of the form*

$$\boldsymbol{X}^* = \boldsymbol{T} \boldsymbol{\Sigma} \boldsymbol{R}^T \qquad \boldsymbol{\Gamma}^* = \boldsymbol{Q} \boldsymbol{T}^{-1} \,, \tag{13}$$

for some matrix $\boldsymbol{T} \in \mathbb{R}^{K \times K}$ *that is commutative in its truncation and inversion.*

Denote the column and row submatrices respectively as $\boldsymbol{A}_b = [T_{1b}, \ldots, T_{(b-1),b}]$ and $\boldsymbol{B}_b = [T_{b1}, \ldots, T_{b,(b-1)}]^T$.

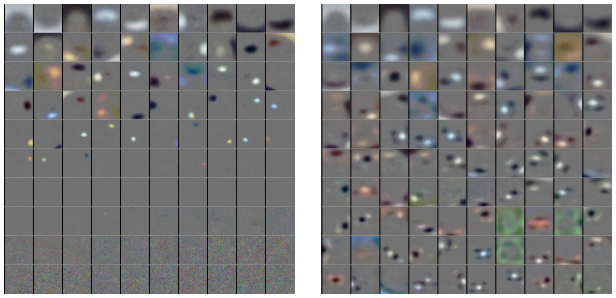
Lemma 1. *Let* $\boldsymbol{T} \in \mathbb{R}^{K \times K}$ *be commutative in its truncation and inversion. Then all the diagonal elements of* $\boldsymbol{T}$ *are nonzero, and for each* $b = 2, \ldots, K$*, either* $\boldsymbol{A}_b = \boldsymbol{0}$ *or* $\boldsymbol{B}_b = \boldsymbol{0}$*.*

In the result below we see that nested dropout coupled with an orthonormality constraint effectively eliminates non-identifiability. The added constraint pins down any possible rotations and scalings.

Theorem 3. *Under the orthonormality constraint* $\boldsymbol{\Gamma}^T \boldsymbol{\Gamma} = \mathbb{I}_K$*, the nested dropout problem features a unique global optimum, and this solution is exactly the set of the* K *top eigenvectors of the covariance of* $\boldsymbol{Y}$*, ordered by eigenvalue magnitude. Namely,* $\boldsymbol{X}^* = \boldsymbol{\Sigma} \boldsymbol{R}^T, \boldsymbol{\Gamma}^* = \boldsymbol{Q}$*.*

4. Training deep models with nested dropout

In this section we discuss our extension of the nested dropout approach to deep architectures. Specifically, we applied this to deep autoencoders having tens of millions of parameters, which we trained on the 80 Million Tiny Images (80MTI) dataset (Torralba et al., 2008) on two GPUs.



(a) Without unit sweeping (b) With unit sweeping

Figure 1. The 100 filters learned by a binarized 3072-100-3072 nested dropout autoencoder on the raw CIFAR-10 pixels where $p_B(\cdot)$ is a geometric distribution with rate of 0.9. For this rate, the probability of sampling any index greater than 50 is ≈ 0.005, and the probability of sampling the 100th unit is ≈ 0.00003: it is very unlikely to ever sample these without unit sweeping. Note the increase of filter fineness as a function of the index.

Training models with nested dropout introduces a number of unconventional technical challenges. In the following sections we describe these challenges, and present strategies to overcome them.

We first describe our general architecture and optimization setup. The 80MTI are 79,302,017 color images of size 32×32. We pre-processed the data by subtracting from each pixel its mean and normalizing by its variance across the dataset. We optimize our models with the nonlinear conjugate gradients algorithm and select step sizes using a strong Wolfe conditions line search. For retrieval-related tasks, we seek to produce binary representations. In light of this we use rectified linear units for all nonlinearities in our encoder, as we find this leads to better binarized representation (see Subsection 4.3). We train for 2 epochs on minibatches of size 10,000. We inject noise to promote robustness, as in (Vincent et al., 2010); namely, with probability 0.1 we independently corrupt input elements to 0. For all layers other than the representation layer, we apply standard dropout with probability 0.2. At each iteration, we sample nested dropout truncation indices for each example in our minibatch, and take a step with respect to the corresponding network mask.

4.1. Unit sweeping for decaying gradients

By the virtue of the decaying distribution $p_B(\cdot)$, it becomes increasingly improbable to sample higher representation indices during training. As such, we encounter a phenomenon where gradient magnitudes vanish as a function of representation unit index. This curvature pathology, in its raw formulation, means that training representation units of higher index can be extremely slow.

In order to combat this effect, we develop a technique we call *unit sweeping*. The idea stems from the observation that the covariance of two latent units sharply de-

creases as a function of the of the difference of their indices. When $p_B(\cdot)$ is a geometric distribution, for example, the probability of observing both units i and j given that one of them is observed is $\mathbb{P}[b \geq \max(i,j) \,|\, b \geq \min(i,j)] = \mathbb{P}[b \geq |i-j|] = \rho^{-|i-j|}$ by the memoryless property of the distribution. In other words, a particular latent unit becomes exponentially desensitized to values of units of higher index. As such, this unit will eventually converge during its training. Upon convergence, then, this unit can be fixed in place and its associated gradients can be omitted. Loosely speaking, this elimination reduces the "condition number" of the optimization. Applying this iteratively, we sweep through the latent units, fixing each once it converges. In Figure 1 we compare filters from training a nested dropout model with and without unit sweeping.

4.2. Adaptive regularization coefficients

The gradient decay as a function of representation index poses a difficulty for regularization. In particular, the ratio of the magnitudes of the gradients of the reconstruction and the regularization vanishes as a function of the index. Therefore, a single regularization term such as $\lambda \sum_{k=1}^{K} \|\mathbf{\Omega}_k\|_{L_1}$ would not be appropriate for nested dropout, since the regularization gradient would dominate the high-index gradients. As such, the regularization strength must be a function of representation index. For weight decay, for example, this would of the form $\sum_{k=1}^{K} \lambda_k \|\mathbf{\Omega}_k\|_{L_1}$. Choosing the coefficients λ_k manually is challenging, and to that end we assign them adaptively. We do this by fixing in advance the ratio between the magnitude of the reconstruction gradient and the regularization gradient, and choosing the λ_k to satisfy this ratio requirement. This corresponds to fixing the relative contributions of the objective and regularizations terms to the gradient for each step of the optimization procedure.

4.3. Code binarization

For the task of retrieval, we would like to obtain binary representations. Several binarization methods have been proposed in prior work (Salakhutdinov & Hinton, 2009; Krizhevsky & Hinton, 2011). We have empirically achieved good performance by tying the weights of the encoder and decoder, and thresholding at the representation layer. Although the gradient itself cannot propagate past this threshold, some signal does: the encoder can be trained since it is linked to the decoder, and its modifications are then reflected in the objective. To attain fixed marginal distributions over the binarized representation units, i.e., $x_k \sim \text{Bern}(\beta)$ for $k = 1, \ldots, K$, we compute the β quantile for each unit, and use this value for thresholding.

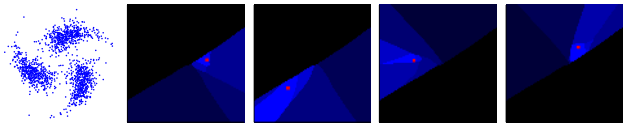


Figure 2. Nested neighborhoods for various examples in a 2D toy problem. The synthetic pinwheel training data is on the top left. The trained model is a 2-16-32-16-2 nested dropout autoencoder. The red dots correspond to the retrieved queries. The shades of blue correspond to different nested neighborhoods for these queries, with color lightness signifying neighborhood depth.

5. Retrieval with ordered binary codes

In this section we discuss how ordered representations can be exploited to construct data structures that permit fast retrieval while at the same time allowing for very long codes.

5.1. Binary tree on the representation space

The ordering property, coupled with the ability to control information capacity decay across representation units, motivates the construction of a binary tree over large data sets. Each node in this tree contains pointers to the set of examples that share the same path down the tree up to that point. Guaranteeing that this tree is balanced is not feasible, as this is equivalent to completely characterizing the joint distribution over the representation space. However, by the properties of the training algorithm, we are able to fix the marginal distributions of all the representation bits as $x_k \sim \text{Bern}(\beta)$ for some hyperparameter $\beta \in (0, 1)$.

Consistent with the training procedure, we encode our database as $\boldsymbol{X} = \{\mathbf{x}_n\}_{n=1}^{N} \subseteq \{0,1\}^K, \mathbf{x}_n = \boldsymbol{f}_{\boldsymbol{\Theta}}(\mathbf{y}_n)$. We then construct a binary tree on the resulting codes.

Given a query $\bar{\mathbf{y}}$, we first encode it as $\bar{\mathbf{x}} = \boldsymbol{f}_{\boldsymbol{\Theta}}(\bar{\mathbf{y}}_n)$. We then conduct retrieval by traveling down the binary tree with each branching determined by the next bit of $\bar{\boldsymbol{x}}$. We define the *b-truncated Hamming neighborhood* of $\bar{\mathbf{x}}$ as the set of all examples whose codes share the first b bits of $\bar{\mathbf{x}}$:

$$N_b^{\mathcal{H}}(\bar{\mathbf{x}}) = \left\{ \mathbf{y} \in \boldsymbol{Y} : \|\mathbf{x}_{\downarrow b} - \bar{\mathbf{x}}_{\downarrow b}\|_{\mathcal{H}} = 0 \right\} . \qquad (14)$$

It is clear that $N_{b+1}^{\mathcal{H}}(\bar{\mathbf{x}}) \subseteq N_b^{\mathcal{H}}(\bar{\mathbf{x}})\ \forall b = 1, \ldots, K-1$. Our retrieval procedure then corresponds to iterating through this family of nested neighborhoods. We expect the cardinality of these to decay approximately exponentially as a function of index. We terminate the retrieval procedure when $\left|N_b^{\mathcal{H}}(\bar{\mathbf{x}})\right| < R$ for some pre-specified terminal neighborhood cardinality, $R \in \mathbb{N}$. It outputs the set $N_{b-1}^{\mathcal{H}}(\bar{\mathbf{x}})$.

Assuming marginals $x_k \sim \text{Bern}(\beta)$ and neglecting dependence between the x_k, this results in expected retrieval time $\mathcal{O}(\frac{\log N/R}{\mathcal{H}(\text{Bern}(\beta))})$ where $\mathcal{H}(\text{Bern}(\beta))$ is the Bernoulli entropy. If $\beta = \frac{1}{2}$, for example, this reduces to the balanced tree travel time $\mathcal{O}(\log N/R)$. This retrieval time is logarithmic in the database size N, and *independent* of the representation space dimensionality K. If one wishes to

retrieve a fixed fraction of the dataset, this renders the retrieval complexity also independent of the dataset size.

In many existing retrieval methods, the similarity of two examples is measured by their Hamming distance. Here, similarity is rather measured by the number of leading bits they share. This is consistent with the training procedure, which produces codes with this property by demanding reconstructive ability under code truncation variation.

5.2. Empirical results

We empirically studied the properties of the resulting codes and data structures in a number of ways. First, we applied ordered retrieval to a toy problem where we trained a tiny 2-16-32-16-2 autoencoder on 2D synthetic pinwheel data (Figure 2). Here we can visualize the nesting of neighborhood families for different queries. Note that, as expected, the nested neighborhood boundaries are orthogonal to the direction of local variation of the data. This follows from the model's reconstruction loss function.

We then trained on 80MTI a binarized nested dropout autoencoder with layer widths 3072-2048-1024-512-1024-2048-3072 with L_1 weight decay and invariance regularization (see Section 4.3). We chose $p_B(\cdot) \sim \text{Geom}(0.97)$ and the binarization quantile $\beta = 0.2$.

Empirical retrieval speeds for various models are shown in Figure 3. We performed retrieval by measuring Hamming distance in a linear scan over the database, and by means of semantic hashing for a number of radii. We also performed ordered retrieval for a number of terminal neighborhood cardinalities. Although semantic hashing is independent of the database size, for a radius greater than 2 it requires more time than a brute force linear scan even for very short codes. In addition, as the code length increases, it becomes very likely that many queries will not find any neighbors for any feasible radii. It can be seen that ordered retrieval carries a very small computational cost which is independent of the code length. Note that each multiplicative variation in the terminal neighborhood size R, from 2 to 32 to 512, leads to a constant shift downward on the logarithmic scale plot. This observation is consistent with our earlier analysis that the retrieval time increases logarithmically with N/R.

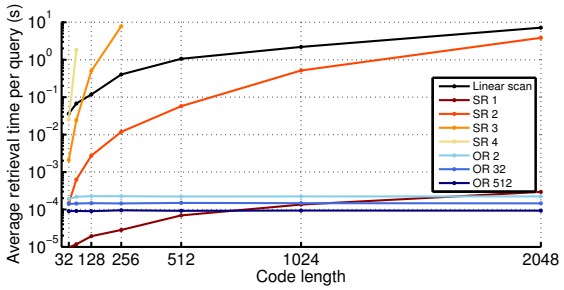


Figure 3. Empirical timing tests for different retrieval algorithms. SR: semantic retrieval, OR: ordered retrieval. The numbers next to "SR" and "OR" in the figure legend correspond to the semantic hashing radius and the terminal ordered retrieval neighborhood cardinality, respectively. As the code length increases, Hamming balls of very small radii become prohibitive to scan. Ordered retrieval carries a small fixed cost that is independent of code length.

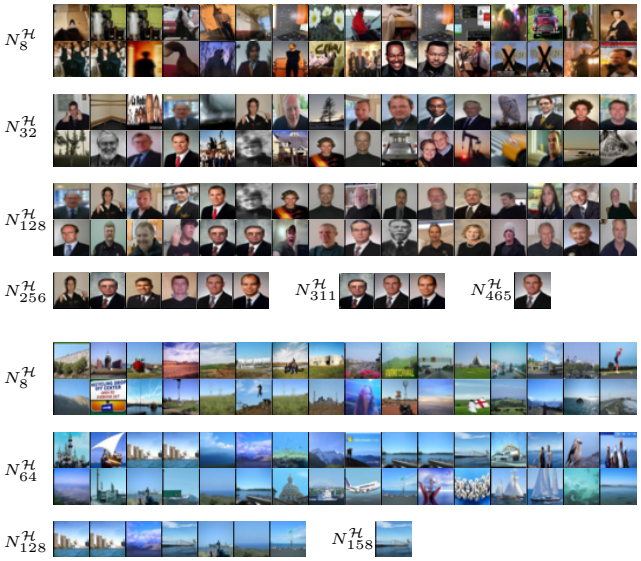


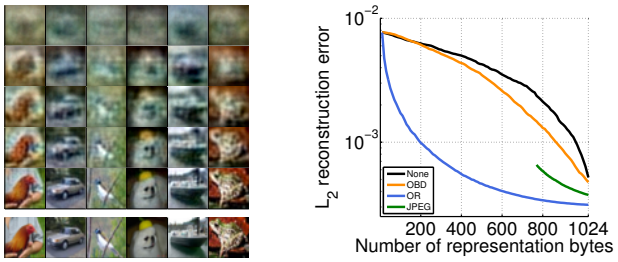
Figure 4. Retrieval results for different terminal neighborhood cardinalities. Note the increase in retrieval fineness as a function of neighborhood index. Examples presented in the order in which they appear in the dataset. When neighborhood sizes are greater than 32, only the first 32 images in the neighborhood are shown. The last neighborhood contains only the query itself.

In Figure 4, we show retrieval results for varying terminal neighborhood sizes. As we decrease the terminal neighborhood size, the similarity of the retrieved data to the query increases. As more bits are added to the representation in the process of retrieval, the resolution of the query increases, and thus it is better resolved from similar images.

6. Adaptive compression

Another application of ordered representations is continuous-degradation lossy compression systems. By "continuous-degradation" we mean that the message can be decoded for any number, b, of bits received, and that the reconstruction error $\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}_{\downarrow b})$ decreases monotonically with b. Such representations give rise to a continuous (up to a single bit) range of bitrate-quality combinations, where each additional bit corresponds to a small incremental increase in quality.

The continuous-degradation property is appealing in many situations. First, consider, a digital video signal that is broadcast to recipients with varying bandwidths. Assume further that the probability distribution over bandwidths for the population, $p_B(\cdot)$, is known or can be estimated, and that a recipient with bandwidth b receives only the first b



(a) Reconstructions (b) Reconstruction rates

Figure 5. Online reconstruction with ordered representations. (a) Reconstructions for code lengths 16, 64, 128, 256, and 1024 using a nested dropout autoencoder. The original images are 24576 bits each. (b) Reconstruction rates as a function of code length for four different truncation techniques: ordered representation, Optimal Brain Damage, a standard autoencoder, and JPEG.

bits of the transmission. We can then pose the following problem: what broadcast signal minimizes the expected distortion over the population? This is formulated as

$$(\mathbf{\Theta}^*, \mathbf{\Psi}^*) = \arg\min_{\mathbf{\Theta},\mathbf{\Psi}} \mathbb{E}_B \left[\mathcal{L}\left(\mathbf{y}_n, \hat{\mathbf{y}}_{n\downarrow b}\right)\right] \ . \qquad (15)$$

This is precisely the optimization problem solved by our model; Equation (15) is simply a rewriting of Equation (3). This connection gives rise to an interpretation of $p_B(\cdot)$, which we have set to the geometric distribution in our experiments. In particular, $p_B(\cdot)$ can be interpreted as the distribution over recipient bandwidths such that the system minimizes the expected reconstruction error.

This intuition in principle applies as well to online video streaming, in which the transmitted signal is destined for only a single recipient. Given that different recipients have different bandwidths, it is acceptable to lower the image quality in order to attain real-time video buffering. Currently, one may specify in advance a small number of fixed encodings for various bandwidths: for example, YouTube offers seven different definitions (240p, 360p, 480p, 720p, 1080p, 1440p, and 2160p), and automatically selects one of these to match the viewer's bitrate. Ordered representations offer the ability to fully utilize the recipient's bandwidth by truncating the signal to highest possible bitrate. Instead of compressing a handful of variants, one needs only to compute the ordered representation once in advance, and truncate it to the appropriate length at transmission time. If this desired bitrate changes over time, the quality could be correspondingly adjusted in a smooth fashion.

6.1. Empirical results

In Figure 5(a), we qualitatively evaluate continuous-degradation lossy compression with ordered representations. We trained a single-layer 3072-1024-3072 autoencoder with nested dropout on CIFAR-10, and produced reconstructions for different code lengths. Each column represents a different image and each row represents a different code length. As the code length increases (downwards in the figure), the reconstruction quality increases. The images second-to-bottom row look very similar to the original uncompressed images in the bottom row (24576 bits each).

Figure 5(b) shows ordered representation reconstruction rates as a function of code length for different approaches to the problem. In addition to the above, we also trained a standard autoencoder with the same architecture but without nested dropout. On this we applied 2 different truncation approaches. The first is a simple truncation on the unordered bits. The second is Optimal Brain Damage truncation (LeCun et al., 1990), which removes units in decreasing order of their influence on the reconstruction objective, measured in terms of the first and second order terms in its Taylor expansion. This is a clever way of ordering units, but is disjoint from the training procedure and is only applied retroactively. We also compare with JPEG compression. We use the libjpeg library and vary the JPEG quality parameter. Higher quality parameters result in larger file sizes and lower reconstruction error. Note that JPEG is not well-suited for the 32x32 pixel images we use in this study; its assumptions about the spectra of natural images are violated by such highly down-sampled images.

7. Discussion and future work

We have presented a novel technique for learning representations in which the dimensions have a known ordering. This procedure is applicable to deep networks, and in the special case of shallow autoencoders is provably exactly equivalent to PCA. This enables learned representations of data that are adaptive in the sense that they can be truncated with the assurance that the shorter codes contain as much information as possible. Such codes are of interest in applications such as retrieval and compression.

The ordered representation retrieval approach can also be used for efficient supervised learning. Namely, it allows performing k-nearest-neighbors on very long codes in logarithmic time in their cardinality. This idea can be combined with various existing approaches to metric learning of kNN and binarized representations (Norouzi et al., 2012; Salakhutdinov & Hinton, 2007; Weinberger & Saul, 2009). The purely unsupervised approaches we have described here have not been empirically competitive with state of the art supervised methods from deep learning. We are optimistic that nested dropout can be meaningfully combined with supervised learning, but leave this for future work.

In addition, ordered representations provide a practical way to train models with an infinite number of latent dimensions, in the spirit of Bayesian nonparametric methods. For example, the distribution $p_B(\cdot)$ can be chosen to have infinite support, while having finite mean and variance.

Acknowledgements This work is partially supported by the Director, Office of Science, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

References

Bourlard, H. and Kamp, Y. Auto-association by multilayer perceptrons and singular value decomposition. Manuscript M217, Philips Research Laboratory, Brussels, Belgium, 1987.

Coates, A., Lee, H., and Ng, A.Y. An analysis of single-layer networks in unsupervised feature learning. In *Proc. of AISTATS*, volume 15, pp. 215–223, 2011.

Datar, Mayur, Immorlica, Nicole, Indyk, Piotr, and Mirrokni, Vahab S. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the Twentieth Annual Symposium on Computational Geometry*, pp. 253–262, New York, NY, USA, 2004. ACM.

Grauman, Kristen and Fergus, Rob. Learning binary hash codes for large-scale image search. In *Machine Learning for Computer Vision*, volume 411 of *Studies in Computational Intelligence*, pp. 49–87. Springer Berlin Heidelberg, 2013.

Hinton, G E and Salakhutdinov, R R. Reducing the dimensionality of data with neural networks. *Science*, 313 (5786):504–507, 2006.

Hinton, Geoffrey E., Srivastava, Nitish, Krizhevsky, Alex, Sutskever, Ilya, and Salakhutdinov, Ruslan. Improving neural networks by preventing co-adaptation of feature detectors. 2012.

Krizhevsky, Alex and Hinton, Geoffrey E. Using very deep autoencoders for content-based image retrieval. In *ESANN*, 2011.

LeCun, Yann and Bengio, Yoshua. *Convolutional Networks for Images, Speech and Time Series*, pp. 255–258. The MIT Press, 1995.

LeCun, Yann, Denker, John S., and Solla, Sara A. Optimal brain damage. In *Advances in Neural Information Processing Systems*, pp. 598–605, 1990.

Norouzi, Mohammad, Fleet, David, and Salakhutdinov, Ruslan. Hamming distance metric learning. In *Advances in Neural Information Processing Systems 25*, pp. 1070–1078. 2012.

Roweis, Sam T. and Saul, Lawrence K. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, December 2000.

Salakhutdinov, Ruslan and Hinton, Geoffrey. Learning a nonlinear embedding by preserving class neighborhood structure. In *Proc. of AISTATS*, volume 11, 2007.

Salakhutdinov, Ruslan and Hinton, Geoffrey E. Semantic hashing. *Int. J. Approx. Reasoning*, 50(7):969–978, 2009.

Tenenbaum, Joshua B., de Silva, Vin, and Langford, John C. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319, 2000.

Torralba, Antonio, Fergus, Robert, and Freeman, William T. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(11): 1958–1970, 2008.

Van der Maaten, Laurens and Hinton, Geoffrey. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(2579-2605):85, 2008.

Vincent, Pascal, Larochelle, Hugo, Lajoie, Isabelle, Bengio, Yoshua, and Manzagol, Pierre-Antoine. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.*, 11:3371–3408, 2010.

Weinberger, Kilian Q. and Saul, Lawrence K. Distance metric learning for large margin nearest neighbor classification. *J. Mach. Learn. Res.*, 10:207–244, June 2009.

Weiss, Yair, Torralba, Antonio, and Fergus, Robert. Spectral hashing. In *NIPS*, pp. 1753–1760, 2008.