On Linear Embeddings and Unsupervised Feature Learning

Ryan Kiros RKIROS@UALBERTA.CA
Csaba Szepesvári SZEPESVA@UALBERTA.CA
University of Alberta, Canada

Abstract

The ability to train deep architectures has led to many developments in parametric, non-linear dimensionality reduction but with little attention given to algorithms based on convolutional feature extraction without backpropagation training. This paper aims to fill this gap in the context of supervised Mahalanobis metric learning. Modifying two existing approaches to model latent space similarities with a Student's t-distribution, we obtain competitive classification performance on CIFAR-10 and STL-10 with k-NN in a 50-dimensional space compared with a linear SVM with significantly more features. Using simple modifications to existing feature extraction pipelines, we obtain an error of 0.40% on MNIST, the best reported result without appending distortions for training.

1. Introduction

The shift towards learning feature hierarchies from sensory input can largely be attributed to the discovery of successful approaches to training deep, multilayer architectures such as the deep belief network (Hinton et al., 2006), stacked auto-associator (Bengio et al., 2007) and energy based models (Ranzato et al., 2006). With respect to dimensionality reduction, these models have allowed for a new approach to learning highly non-linear, parametric embeddings. Weston et al. (2008) considered deep, semi-supervised models of manifold learning algorithms such as Isomap and Laplacian eigenmaps, while Salakhutdinov & Hinton (2007) adapted their neighbourhood components analysis (NCA) (Goldberger et al., 2004) objective to be used with a deep pre-trained network. Other approaches that were adapted to these architectures include parametric t-SNE (van der Maaten, 2009), large-margin k-NN (Min et al., 2009) and variations of NCA and maximally collapsing metric learning (MCML) (Min et al., 2010) that followed van der Maaten (2009) using heavy tailed distributions for modelling latent similarities. All of these algorithms led to embeddings of far superior quality then that of their linear counterparts.

Although much of the motivation for learning useful feature representations were based on constructing deep architectures, Coates et al. (2011) showed that single layer networks can perform surprisingly well in object classification tasks. This was followed with Coates & Ng (2011a) who argued that encodings are far more important than training dictionaries and attributed the encoder for much of the success of sparse coding. Simple models that required no tuning, such as Ngiam et al. (2011), could be competitive with models such as RBMs that were much more difficult to train. Since then, hierarchical methods have been proposed using much simpler feature learning approaches such as K-SVD (Bo et al., 2011b) and stacked ISA (Le et al., 2011).

The aim of this paper is to consider dimensionality reduction in architectures with convolutional feature extraction and no backpropagation, showing that simple linear embeddings can be successfully learned on top of these extracted representations. We utilize two existing Mahalanobis metric learners, correlative matrix mapping (Strickert et al., 2010) (CMM) and maximally collapsing metric learning (Globerson & Roweis, 2006) (MCML) and introduce mini batch training where latent representations are modelled using a Student's t-distribution. To further motivate this setting, we describe a generalized two-step procedure for performing non-linear dimensionality reduction, for which spectral methods fall under. We experiment on three datasets, MNIST, CIFAR-10 and STL-10 learning 50-dimensional embeddings on each using both fully supervised and semi-supervised settings. Using a k-NN classifier in the latent space, we obtain com-

petitive classification performance on CIFAR-10 and STL-10 as well as the best classification performance reported on MNIST when no distortions are appended for training.

2. Methods

We split our discussion into three sections: the pipeline for unsupervised feature learning, descriptions of the embedding procedure and finally motivation for our 2-step approach. We chose to largely follow the pipeline of Coates & Ng (2011a) due to its simplicity and previous success in object recognition. Next, motivated by the supervised t-distributed embeddings of Min et al. (2010), we modify CMM and MCML and model latent distributions with a Student's t-distribution. Finally, we discuss the similarities of our approach to spectral dimensionality reduction and describe a generalized strategy for non-linear dimensionality reduction. Our goal is then to show that under this setting, k-NN classification performance in a low-dimensional space can be competitive with a linear SVM with thousands more features.

2.1. Unsupervised Feature Learning

Our pipeline for representation learning mostly follows Coates & Ng (2011a) (contrast normalization, whitening and convolutional feature extraction) along some modifications based on Jarrett et al. (2009) (tanh activation and contrast normalization) as well as the use of a 2-layer average pooling. We assume that we are given a set of image patches $P = \{p^{(1)}, \ldots, p^{(m)}\}$ of size $r \times c$ where $r \times c$ is the receptive field size. The patches are extracted randomly across all images and will be used for performing dictionary learning. Each patch $p^{(i)}$ is then collapsed into a single vector of size $3rc$ (for color images) and rc for grayscale images. For color patches each component is left unnormalized while for grayscale patches we divide by 255 to map the data into [0,1].

We perform an additional pre-processing step for color patches by subtracting the mean and dividing by the standard deviation across patches. These operations correspond to brightness and contrast normalization.

2.1.1. Whitening

Whitening is an operation that is used to decrease intensity correlation between neighbouring pixels. More specifically, after whitening, patches have zero mean, $\sum_{i=1}^{m} p^{(i)} = 0$, and identity covariance, $\frac{1}{m}\sum_{i=1}^{m} p^{(i)}(p^{(i)})^T = I$. We make use of zero-phase (ZCA) whitening which is performed by first subtracting the mean, performing an eigendecomposition $C = VDV^T$ on the covariance matrix $C = C(P)$ then computing the whitening matrix $W = V(\Delta(\delta(D) + \epsilon))^{-\frac{1}{2}}V^T$ where ϵ is a small positive number applied to the diagonal of D. The use of ϵ has the effect of low-pass filtering the data if set sufficiently high. For all of our experiments we set $\epsilon = 0.1$ as is done in Coates & Ng (2011a).

2.1.2. Dictionary Learning

After performing whitening, the patches are now ready to be used for constructing a dictionary. We follow Coates & Ng (2011a) and use orthogonal matching persuit (OMP). OMP aims to solve the following optimization problem:

$$\begin{aligned} \underset{D,\hat{p}^{(i)}}{\text{minimize}} \quad & \sum_{i=1}^{m} ||D\hat{p}^{(i)} - p^{(i)}||_2^2 \\ \text{subject to} \quad & ||D^{(j)}||_2^2 = 1, \forall j \\ & ||\hat{p}^{(i)}||_0 \leq q, \forall i \end{aligned} \tag{1}$$

where $D \in \mathbb{R}^{n \times k}$ and $D^{(j)}$ is the j-th column of D. The objective is minimized using alternation with $q = 1$.

Recently, Coates & Ng (2011a) showed that using randomly chosen patches as a dictionary can be surprisingly effective. Thus, we also consider constructing a dictionary by simply choosing a random subset of $P = \{p^{(1)}, \ldots, p^{(m)}\}$ to be the columns of D, followed by normalization with $||D^{(j)}||_2^2 = 1$. Zhang et al. (2011) showed that under a convex relaxation of sparse coding, global solutions can be obtained simply from normalizing over examples. This observation might help explain why such bases often perform well in practice.

2.1.3. Convolutional Feature Extraction

Let $\{I^{(1)}, \ldots, I^{(m)}\} \in \mathbb{R}^{n_V \times n_H}$ represent a set of input images of dimensions $n_V \times n_H$ for which we would like to map to a new feature representation using the learned bases D. Suppose that the receptive field size is $r \times c$ and that the j-th subregion (patch) of $I^{(i)}$ is $s_j^{(i)}$ collapsed into a column vector. Then the feature encoding $f_j^{(i)}$ is given by $f_j^{(i)} = \max\{\tanh(D^T s_j^{(i)}) - \alpha, 0\}$ where α is a parameter to be set by the user. When the stride between image subregions is one, the result is equivalent to a valid convolution with the dictionary. In particular, the resulting encodings $f^{(i)}$ for image $I^{(i)}$ will be of size $(n_V - r + 1) \times (n_H - c + 1) \times k$, where k is the

size of the dictionary D. Given the encodings $f^{(i)}$, we perform one additional step of contrast normalization of the form $\hat{f}^{(i)} = (f^{(i)} - \mu(f^{(i)}))/\max\{\mu(\sigma), \sigma^{(i)}\}$, where μ, σ denotes vectors of means and standard deviations across patches. The use of the tanh activation and contrast normalization deviates from the pipeline of Coates & Ng (2011a) but was shown to be useful operations by Jarrett et al. (2009) when analyzing different pipelines for object recognition tasks.

2.1.4. Pooling

The final procedure in our pipeline is performing pooling over the extracted encodings. We utilize a 2 layer pyramid consisting of an initial 4×4 average pooling over the extracted encodings $\hat{f}^{(i)}$. This is followed by splitting the output into quadrants and average pooling over each quadrant. The final output is of size $2 \times 2 \times k$ for which features are then flattened into a single feature vector of size $4k$. To get the final feature vector $x^{(i)}$ for image $I^{(i)}$ to use as input for dimensionality reduction, we standardize the data using mean centring and regularized variance normalization $x_j^{(i)} = (\hat{f}_j^{(i)} - \mu_j)/\sqrt{\sigma_j^2 + \epsilon}$ for feature means μ_j and standard deviations σ_j with $\epsilon = 0.01$ for all experiments.

2.1.5. The Role of Learning

Our pipeline may be summarized as follows: initial contrast normalization, whitening, dictionary learning, convolutional feature extraction, additional contrast normalization and pooling. Under this setup, the only learning that takes place is when OMP is used to construct a dictionary. If we instead construct a dictionary just from normalizing over randomly selected patches, then no learning takes place at all. Thus in the minimal setting, only the linear embedding of the extracted features is learned, excluding the use of k-NN for classification.

2.2. Dimensionality Reduction

2.2.1. Correlative Matrix Mapping

Correlative matrix mapping (CMM) (Strickert et al., 2010) learns a Mahalanobis type metric such that latent distances are in maximum correlation with label distances. We modify this existing approach by modelling the latent representations using a Student's t-distribution. Let $X = \{x^{(1)} \dots x^{(m)}\}$ be a set of m datapoints (in our case, the extracted features) and let $l^{(i)}$ denote the associated label of $x^{(i)}$ which we shall assume is a binary vector of length $|c|$ where c is the set of all single-label classes. The vector $l^{(i)}$ has a 1 in the k^{th} component if it belongs to class k and 0 otherwise. Let $f(x^{(i)}|W) = Wx^{(i)}$ with $d_{ij} = \|f(x^{(i)}|W) - f(x^{(j)}|W)\|_2$. Consider a joint distribution Q with entries q_{ij} representing the probability that $x^{(i)}$ would have $x^{(j)}$ as a neighbour under a Student's t-distribution with α degrees of freedom:

$$q_{ij} = \frac{(1 + d_{ij}^2/\alpha)^{\frac{-\alpha+1}{2}}}{\sum_{k \neq l}(1 + d_{kl}^2/\alpha)^{\frac{-\alpha+1}{2}}}, \qquad q_{ii} = 0 \tag{2}$$

Let D_L denote a joint distribution of the label similarities, where $(D_L)_{ij}$ is the normalized cosine similarity between $l^{(i)}$ and $l^{(j)}$ such that $\sum_{ij}(D_L)_{ij} = 1$. Our objective can now be expressed as:

$$C_{t-CMM} = \max_W r(D_L, Q) - \beta\|W\|_F^2 \tag{3}$$

where r denotes Pearson's correlation. We follow van der Maaten (2009) and learn the degrees of freedom by initializing it to $\alpha = d - 1$ and computing $\frac{\partial C}{\partial \alpha}$ at each update. We will denote the modified version of CMM by t-CMM.

2.2.2. Maximally Collapsing Metric Learning

Maximally collapsing metric learning (MCML) (Globerson & Roweis, 2006) also aims to learn a Mahalanobis type metric, although the motivations are different. MCML collapses classes with the intention that all points in the same class should be mapped to the same location. As with CMM, the objective is modified to be applied for use with a Student's t-distribution. Let q_{ij} and d_{ij} be defined as in Equation 2. Consider the following joint distribution over the label space:

$$p_{ij} \propto \begin{cases} 1 & \text{if } y^{(i)} = y^{(j)} \\ 0 & \text{if } y^{(i)} \neq y^{(j)} \end{cases} \tag{4}$$

the modified objective can now be expressed as:

$$C_{t-MCML} = \min_W \sum_i \sum_{j \neq i} p_{ij} \log \frac{p_{ij}}{q_{ij}} + \beta\|W\|_F^2 \tag{5}$$

We found that the use of regularization for both t-CMM and t-MCML always improved performance in our experiments. We also note that the non-regularized MCML formulation can be seen as a special case of Min et al. (2010) when only a single layer with a linear activation is used for the embedding.

2.2.3. Optimization

Due to the quadratic complexity of our objectives, running the algorithms on more then 10000+ datapoints becomes both memory and computationally inefficient. We instead propose to use minibatch training, as has been done by Salakhutdinov & Hinton (2007) among others. Optimization is done by running L-BFGS for 10 iterations on each minibatch, with a full epoch representing a full pass through the training set. Due to the non-convexity of the objectives, we also briefly experimented with various number of iterations, batch sizes as well as the use of applying conjugate gradients as opposed to L-BFGS. We found that the results were indifferent so long as the number of iterations per minibatch was sufficiently high.

2.3. Motivation: Spectral Dimensionality Reduction

Our procedure of first performing representation learning followed by a linear embedding can be generalized in the following setting:

1. Learn a non-linear representation defined by $\phi : \mathbb{R}^p \to \mathbb{R}^q$ which may or may not depend on the desired dimension d
2. Construct a linear embedding $\gamma : \mathbb{R}^q \to \mathbb{R}^d$ of the output representations.

We first observe that spectral methods for dimensionality reduction fall into this framework. Let $X = \{x^{(1)}, \ldots, x^{(m)}\}$ be a set of m datapoints with arbitrary centred kernel matrix $K \in \mathbb{R}^{m \times m}$. Consider the following objective:

$$\min_{\bar{K}=\bar{K}^T, \bar{K} \succeq 0} L(D(\bar{K}); D(K)) + R(\bar{K}) \tag{6}$$

where $D(K)$ is the distance matrix representation of K, $\bar{K}$ is a reconstruction of K and L, R are a specified loss and regularizer, respectively. Given an optimal solution, one can obtain a representation $\hat{X}$ by performing an eigendecomposition $\bar{K} = QVQ^T$ and truncating all but the top d eigenvalues, where d is the desired dimensionality of the embedding.

Under this framework, set $\phi(K) = \operatorname{argmin}_{\bar{K}=\bar{K}^T, \bar{K} \succeq 0} L(D(\bar{K}); D(K)) + R(\bar{K})$ and $\gamma(\bar{K}) = \hat{X}$ such that the desired embedding $\hat{X}$ is given by $(\gamma \circ \phi)(K)$ with $p = q = m \times m$ and domain of ϕ the set of valid kernels. Although the actual feature representations might be of infinite size implicitly represented by a choice of kernel, we distinguish this by the explicit representation learned in terms of the kernel.

2.3.1. Example: Kernel PCA

In the case of kernel PCA, we use $L(\bar{D}; D) = \|H(D - \bar{D})H\|$ where H is a mean centering matrix and $R(\bar{K}) = [[\text{rank}(\bar{K}) \leq d]]$. In this example, the representation learning phase depends on d to enforce low rank in $\bar{K}$.

2.3.2. Example: Maximum Variance Unfolding (MVU)

For MVU (Weinberger & Saul, 2004), we set $L(\bar{D}; D) = \sum_{ij}[[N(D)_{ij} = 1 \text{ and } \bar{D}_{ij} \neq D_{ij}]]$ where $N(D_{ij}) = 1$ if $x^{(i)}$ and $x^{(j)}$ are neighbours in D and 0 otherwise. The regularizer used is of the form $R(\bar{K}) = -\text{tr}(\bar{K})$.

2.3.3. Generalizations

Our proposed framework for embedding images we let our mapping be defined independently of d to be that of our feature extraction pipeline where the mapping is parametrized by a constructed dictionary. The embedding phase is then a straightforward rank d linear transformation of the output features. In this paper, we restrict ourselves to learning supervised Mahalanobis metrics parametrized by the embedding matrix. Observe that self-taught learning naturally falls into this framework by setting ϕ to be the feature map learned from an arbitrary image database, potentially independent of the desired recognition task.

3. Experiments

For our experiments we use 3 datasets commonly used in the deep learning/feature learning community: MNIST [1], CIFAR-10 [2] (Krizhevsky, 2010) and STL-10 [3] (Coates et al., 2011).

MNIST is a collection of 70000 28×28 grayscale images of digits from 0-9. The dataset is partitioned into 60000 for training and 10000 for testing. CIFAR-10 is a collection of 60000 32×32 color images of objects from 10 classes, each which is evenly distributed across the training and testing sets. The training set is partitioned into 50000 images with the test set having 10000. STL-10 is a recent dataset used to motivate further work in self-taught learning. STL-10 contains

[1] http://yann.lecun.com/exdb/mnist/
[2] http://www.cs.toronto.edu/~kriz/cifar.html
[3] http://cs.stanford.edu/~acoates/stl10

10 folds of size 1000 each of 96×96 images from 10 classes. It also includes an addition unlabelled set of 100000 images that include images from distributions other than the intended classification task. The procedure to using the dataset is to first perform representation learning on the unlabelled dataset, then apply the learned features to each of the 10 training folds, reporting the average classification performance on a test set of size 8000.

For all experiments, we use the described pipeline for learning a feature representation from the input images, then utilize either t-CMM or t-MCML for performing the supervised embeddings. We use a receptive field size of 9×9, 4×4 + quadrant pooling and 4000 bases on all datasets. On MNIST, we consider experiments with 2 bases: learned (**P**) and random patches (**RP**). For the other 2 datasets, we use 3: learned (**P**), random patches (**RP**) and learned with both datasets (**D**). We also consider semi-supervised experiments where 1%, 5% and 10% of the data is used for each of the above combination of trials. This corresponds to using only those percentages of data for the supervised embeddings, while all the data is still used for unsupervised feature learning.

For performing the embeddings, we partition the training set for training and validation and run the embedding algorithm until the objective value does not improve on the validation set for 10 epochs. All experiments are run with minibatches of size no larger than 3000. A value of k from 1,3,5,10,15,20 and 25 is then chosen that maximizes the classification accuracy on the validation set. Once this is completed, the algorithm is ran again using the full training set for the indicated number of epochs. Classification is performed using the chosen value of k from the validation set. In all of our experiments, we fix the regularization parameter to $\beta = 0.01$ and soft-activation parameter $\alpha = 0.25$.

In some of our experiments, we found the above procedure led to underfitting, in the sense that although the validation objective was no longer improving, the classification performance was still improving. In these cases, we simply removed the early stopping criteria and ran the algorithm for a total of 50 epochs. A value of k is chosen as before, then the algorithm is applied again for 50 epochs on the full training set. All experiments on CIFAR-10, as well as the fully-supervised MNIST experiments fell into this category. All STL-10 experiments benefited from the early stopping.

For experimental control, we only compare our methods with similar non-backprop pipelines when no distortions are added for training. For additional comparison, we trained a linear L2-SVM on the fully supervised experiments with basis **P** in order to gauge the modifications made to the feature extraction pipeline. Model selection is performed with the same validation sets as the embedding procedures. Code for reproducing our results will be made available online.

Method (% of labels)	P	RP	D
t-CMM (1%)	49.18	47.27	49.75
t-MCML	50.25	48.40	50.46
t-CMM (5%)	61.84	60.95	62.17
t-MCML	64.01	63.14	64.31
t-CMM (10%)	67.01	66.39	68.09
t-MCML	68.72	67.66	69.68
t-CMM (100%)	79.44	78.21	79.78
t-MCML	78.69	78.69	**80.12**

Table 1. k-NN classification accuracy of the learned 50 dimensional features on the CIFAR-10 dataset using various dictionaries and label percentages.

Method	Accuracy	Features
Coates & Ng (2011a)	81.5%	48000
Jia et al. (2012)	**83.1%**	24000
Coates & Ng (2011b)	82.0%	22400
Coates et al. (2011)	79.6%	16000
Bo et al. (2011a)	80.0%	6000
SVM	80.0%	16000
t-MCML + k-NN	80.1%	50

Table 2. A selection of the best results obtained on CIFAR-10 using non-backprop pipelines, sorted by number of features used.

3.1. CIFAR-10

The results obtained on CIFAR-10 are in Table 1. The best results are obtained with the inclusion of the STL-10 unlabelled data for dictionary learning. Table 2 shows a selection of the best results obtained from the literature using non-backprop pipelines, sorted by the number of features used for classification. Our best accuracy of 80.1% is competitive with all existing methods. Moreover, our results show that having thousands of features is not a necessity for good classification performance. We note that other approaches based on backpropagation convolutional networks exist (Krizhevsky, 2010; Cireşan et al., 2011; 2012), the latter being the current state-of-the-art on this dataset which appended added distortions for training. To further test the quality of the learned embeddings, we applied t-SNE to a random subset of 2500 test datapoints and visualized the two dimensional embeddings in Figure 1. Interestingly, all of the man-made objects (air-

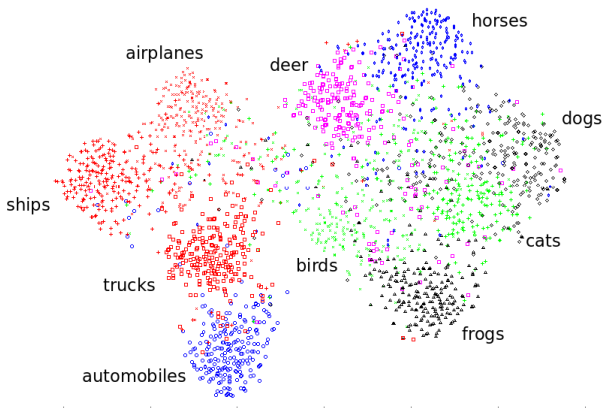

Figure 1. t-SNE embedding of the learned 50 dimensional features on a random subset of 2500 CIFAR-10 test datapoints. All the training data was used for learning both the dictionary and embedding (best seen in color)

Method	P	RP	D
t-CMM	53.77 ± 1.29	52.88 ± 1.15	**54.23 ± 0.73**
t-MCML	51.82 ± 3.45	44.84 ± 4.33	49.18 ± 3.54

Table 3. k-NN classification accuracy and standard deviations of the learned 50 dimensional features on the STL-10 dataset using various dictionaries.

planes, automobiles, ships and trucks) are clustered off on one side while living creatures (birds, cats, deer, dogs, frogs and horses) are clustered on the other end. The birds class is centred due to its similarity with some of the airplanes.

3.2. STL-10

The results of our experiments on STL-10 are shown in Table 3. As in the case of CIFAR-10, the best result is obtained by using both datasets for dictionary learning. Surprisingly, we found that t-MCML often gave very poor results from many of the folds. This behaviour was not observed with t-CMM and is consistent across all dictionaries. Table 4 shows how our performance compares with existing methods in the literature. Our best result is roughly on par with that of Coates & Ng (2011a) when a soft activation is used for encoding. Improved performance was also obtained by replacing the soft-activation with the sparse coding encoder, explaining that on smaller labelled sets the use of the SC encoder is preferable. The best approaches on this dataset are obtained using features learned through temporal coherence and a 3 layer network with receptive field learning in the higher layers. As with the CIFAR-10 results, the number of features used is significantly less than that of existing methods.

Method	Accuracy	Features
Coates & Ng (2011b)	60.1%	22400
Coates & Ng (2011a)	54.9%	12800
Coates & Ng (2011a)	59.0%	12800
Coates et al. (2011)	51.5%	6400
Ngiam et al. (2011)	53.5%	6400
Zou et al. (2011)	**61.0%**	-
SVM	58.0%	16000
t-MCML + k-NN	54.2%	50

Table 4. A selection of the best results obtained on STL-10 using non-backprop pipelines, sorted by number of features used.

Method (% of labels)	P	RP
t-CMM (1%)	2.51	2.98
t-MCML	2.60	2.74
t-CMM (5%)	1.05	1.23
t-MCML	1.06	1.06
t-CMM (10%)	0.79	0.87
t-MCML	0.93	0.80
t-CMM (100%)	0.46	0.52
t-MCML	**0.44**	0.51

Table 5. k-NN classification errors of the learned 50 dimensional features on the MNIST dataset using various dictionaries and label percentages.

Method	Error
Ranzato et al. (2007)	0.62%
Ranzato et al. (2006)	0.60%
Labusch et al. (2008)	0.59%
Keysers et al. (2007)	0.54%
Jarrett et al. (2009)	0.53%
Keysers et al. (2007)	0.52%
t-CMM + k-NN	0.46%
t-MCML + k-NN	0.44%
SVM	**0.40%**

Table 6. A selection of the best results obtained on MNIST when no distortions are added to the training set.

3.3. MNIST

Finally, we performed experiments on the MNIST dataset for which the classification errors of are shown in Table 5. For additional comparison of our semi-supervised results, Lee et al. (2009) obtain a result of 1.91% error with a convolutional deep belief network and Weston et al. (2008) a result of 1.83% using 5% of labelled data, where our best result of 1.05% sig-

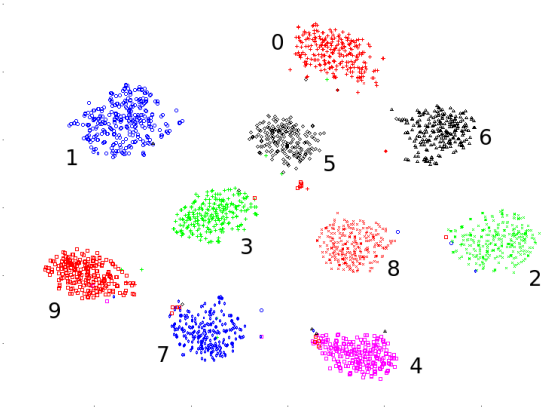

Figure 2. t-SNE embedding of the learned 50 dimensional features on a random subset of 2500 MNIST test datapoints, when only 1% of labelled training data was used to learn the t-CMM embedding. All the training data was used for unsupervised learning of the dictionary (best seen in color)

nificantly outperforms these methods. Table 6 shows a comparison of the best results obtained on MNIST when no distortions are added to the training set. For additional comparison, we trained a linear L2-SVM on the high dimensional features which is the best result obtained on this dataset [4]. As with CIFAR-10, we test how well our learned embeddings can be visualized in a 2 dimensional space. We selected a random subset of 2500 test datapoints and ran t-SNE on the learned 50 dimensional embeddings. t-SNE is able to naturally separate clusters when only 1% of labelled data is used to perform the embedding on the training data (Figure 2).

4. Conclusion

In this paper we show that, parallel to the success of linear classifiers, simple linear embeddings can be successfully used to reduce the dimension of features convolutionally extracted from images. Our partitioned approach is motivated by the same two-step approach used in spectral dimensionality reduction. Using small modifications to correlative matrix mapping and maximally collapsing metric learning, we obtain classification results in 50 dimensions that are competitive with state-of-the-art approaches that uses thousands of features. In the case of MNIST, we improve on the current state-of-the-art when no distortions are added to the training set. Moreover, in a minimalist setting, the only learning necessary is the embedding excluding the k-NN classifier.

[4]Based on results posted on `http://yann.lecun.com/exdb/mnist/`

Although throughout our experiments we use OMP for dictionary learning and a soft-threshold for activation, one could replace these with any other training and encoding procedure. In particular, it is conceivable that the use of the sparse coding encoder on STL-10, as was done in Coates & Ng (2011a), could further improve the embedding performance. Furthermore, the same could hold true when more layers are added with a receptive field learning procedure of Coates & Ng (2011b) and Jia et al. (2012). The use of scalable algorithms that can learn representations at the pixel level for full sized images, such as hierarchical sparse coding (Yu et al., 2011) and hierarchical matching pursuit (Bo et al., 2011b) could be applied in combination with our linear embeddings for learning low dimensional representations of full sized images.

Future work in this direction would focus more on unsupervised embeddings. As an example, one could test the ability to learn binary codes on full sized images for semantic hashing (Hinton & Salakhutdinov, 2011) but replacing a deep autoencoder with one that has only a single hidden layer. Finally, one could extend the use of self-taught learning to both the representation learning phase and embedding phase.

5. Acknowledgments

The authors would like to thank Dale Schuurmans as well as the reviewers for helpful comments and suggestions.

References

Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. Greedy layer-wise training of deep networks. *NIPS*, 19:153, 2007.

Bo, L., Lai, K., Ren, X., and Fox, D. Object recognition with hierarchical kernel descriptors. In *CVPR*, pp. 1729–1736. IEEE, 2011a.

Bo, L., Ren, X., and Fox, D. Hierarchical Matching Pursuit for Image Classification: Architecture and Fast Algorithms. In *NIPS*, December 2011b.

Cireşan, D., Meier, U., and Schmidhuber, J. Multi-column deep neural networks for image classification. *Arxiv preprint arXiv:1202.2745*, 2012.

Cireşan, D.C., Meier, U., Masci, J., Gambardella, L.M., and Schmidhuber, J. High-performance neural networks for visual object classification. *Arxiv preprint arXiv:1102.0183*, 2011.

Coates, A. and Ng, A.Y. The importance of encod-

ing versus training with sparse coding and vector quantization. In *ICML*, volume 8, pp. 10, 2011a.

Coates, A., Lee, H., and Ng, A. Y. An analysis of single-layer networks in unsupervised feature learning. In *AISTATS*, 2011.

Coates, Adam and Ng, Andrew Y. Selecting receptive fields in deep networks. *NIPS*, (i):1–9, 2011b.

Globerson, A. and Roweis, S. Metric learning by collapsing classes. *NIPS*, 18:451, 2006.

Goldberger, J., Roweis, S., Hinton, G., and Salakhutdinov, R. Neighbourhood components analysis. 2004.

Hinton, G. and Salakhutdinov, R. Discovering binary codes for documents by learning deep generative models. *Topics in Cognitive Science*, 3(1):74–91, 2011.

Hinton, G.E., Osindero, S., and Teh, Y.W. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

Jarrett, Kevin, Kavukcuoglu, Koray, Ranzato, Marc Aurelio, and LeCun, Yann. What is the best multi-stage architecture for object recognition? *ICCV*, 6:2146–2153, 2009.

Jia, Y., Huang, C., and Darrell, T. Beyond spatial pyramids: Receptive field learning for pooled image features. In *CVPR*. IEEE, 2012.

Keysers, D., Deselaers, T., Gollan, C., and Ney, H. Deformation models for image recognition. *PAMI*, 29(8):1422–1435, 2007.

Krizhevsky, A. Convolutional deep belief networks on cifar-10. *Unpublished manuscript*, 2010.

Labusch, K., Barth, E., and Martinetz, T. Simple method for high-performance digit recognition based on sparse coding. *IEEE Transactions on Neural Networks*, 19(11):1985–1989, 2008.

Le, Q.V., Zou, W.Y., Yeung, S.Y., and Ng, A.Y. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *CVPR*, pp. 3361–3368. IEEE, 2011.

Lee, Honglak, Grosse, Roger, Ranganath, Rajesh, and Ng, Andrew Y. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. *ICML*, pp. 1–8, 2009.

Min, R., Stanley, D.A., Yuan, Z., Bonner, A., and Zhang, Z. A deep non-linear feature mapping for large-margin knn classification. In *Ninth IEEE International Conference on Data Mining*, pp. 357–366. IEEE, 2009.

Min, R., van der Maaten, L., Yuan, Z., Bonner, A., and Zhang, Z. Deep supervised t-distributed embedding. In *ICML*, volume 1, 2010.

Ngiam, Jiquan, Koh, Pang Wei, Chen, Zhenghao, Bhaskar, Sonia, and Ng, Andrew Y. Sparse filtering. *NIPS*, 25:1–9, 2011.

Ranzato, M., C.P., Chopra, S., and LeCun, Y. Efficient learning of sparse representations with an energy-based model. *NIPS*, 19:1137–1144, 2006.

Ranzato, M.A., Huang, F.J., Boureau, Y.L., and Lecun, Y. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *CVPR*, pp. 1–8. IEEE, 2007.

Salakhutdinov, R. and Hinton, G. Learning a nonlinear embedding by preserving class neighbourhood structure. In *AI and Statistics*, volume 3, pp. 5, 2007.

Strickert, M., Soto, A.J., and Vazquez, G.E. Adaptive matrix distances aiming at optimum regression subspaces. In *ESANN*, pp. 93–98, 2010.

van der Maaten, L. Learning a parametric embedding by preserving local structure. *Proceedings of AISTATS*, 2009.

Weinberger, K.Q. and Saul, L.K. Unsupervised learning of image manifolds by semidefinite programming. In *CVPR*, volume 2, pp. II–988. IEEE, 2004.

Weston, J., Ratle, F., and Collobert, R. Deep learning via semi-supervised embedding. In *ICML*, pp. 1168–1175. ACM, 2008.

Yu, K., Lin, Y., and Lafferty, J. Learning image representations from the pixel level via hierarchical sparse coding. In *CVPR*, pp. 1713–1720. IEEE, 2011.

Zhang, X., Yu, Y., White, M., Huang, R., and Schuurmans, D. Convex sparse coding, subspace learning, and semi-supervised extensions. In *Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.

Zou, W., Ng, A., and Yu, Kai. Unsupervised learning of visual invariance with temporal coherence. In *NIPS*, 2011.