Improved Local Coordinate Coding using Local Tangents

Kai Yu KYU@SV.NEC-LABS.COM
NEC Laboratories America

Tong Zhang TZHANG@STAT.RUTGERS.EDU
Rutgers University

Abstract

Local Coordinate Coding (LCC), introduced in (Yu et al., 2009), is a high dimensional nonlinear learning method that explicitly takes advantage of the geometric structure of the data. Its successful use in the winning system of last year's Pascal image classification Challenge (Everingham, 2009) shows that the ability to integrate geometric information is critical for some real world machine learning applications. This paper further develops the idea of integrating geometry in machine learning by extending the original LCC method to include local tangent directions. These new correction terms lead to better approximation of high dimensional nonlinear functions when the underlying data manifold is locally relatively flat. The method significantly reduces the number of anchor points needed in LCC, which not only reduces computational cost, but also improves prediction performance. Experiments are included to demonstrate that this method is more effective than the original LCC method on some image classification tasks.

1. Introduction

This paper considers the problem of learning a nonlinear function $f(x)$ in high dimension: $x \in \mathbb{R}^d$ with large d. We are given a set of labeled data $(x_1, y_1), \ldots, (x_n, y_n)$ drawn from an unknown underlying distribution. Moreover, we assume that an additional set of unlabeled data $x \in \mathbb{R}^d$ from the same distribution may be observed.

If the dimensionality d is large compared to n, then the traditional statistical theory predicts over-fitting due to the so called "curse of dimensionality". However, for many real problems with high dimensional data, we do not observe this so-called curse of dimensionality. This is because although data are physically represented in a high-dimensional space, they often lie (approximately) on a manifold which has a much smaller intrinsic dimensionality.

A new learning method, called Local Coordinate Coding or LCC, was recently introduced in (Yu et al., 2009) to take advantage of the manifold geometric structure to learn a nonlinear function in high dimension. The method was successfully applied to image classification tasks. In particular, it was the underlying method of the winning system for the Pascal image classification challenge last year (Everingham, 2009). Moreover, that system only used simple SIFT features that are standard in the literature, which implies that the success was due to the better learning method rather than better features. The reason for LCC's success for image classification is due to its ability to effectively employ geometric structure which is particularly important in some real applications including image classification.

The main idea of LCC, described in (Yu et al., 2009), is to locally embed points on the underlying data manifold into a lower dimensional space, expressed as coordinates with respect to a set of anchor points. The main theoretical observation was relatively simple: it was shown in (Yu et al., 2009) that on the data manifold, a nonlinear function can be effectively approximated by a globally linear function with respect to the local coordinate coding. Therefore the LCC approach turns a very difficult high dimensional nonlinear learning problem into a much simpler linear learning problem, which can be effectively solved using standard machine learning techniques such as regularized linear classifiers. This linearization is effective because the method naturally takes advantage of the geometric in-

Appearing in *Proceedings of the 26th International Conference on Machine Learning*, Haifa, Israel, 2010. Copyright 2010 by the author(s)/owner(s).

formation.

However, LCC has a major disadvantage, which this paper attempts to fix. In order to achieve high performance, one has to use a large number of so-called "anchor points" to approximate a nonlinear function well. Since the "coding" of each data point x requires solving a Lasso problem with respect to the anchor points, it becomes computationally very costly when the number of anchor points becomes large.

Note that according to (Yu et al., 2009), the LCC method is a local linear approximation of a nonlinear function. For smooth but highly nonlinear functions, local linear approximation may not necessarily be optimal, which means that many anchor points are needed to achieve accurate approximation. This paper considers an extension of the local coordinate coding idea by including quadratic approximation terms. As we shall see, the new terms introduced in this paper correspond to local tangent directions.

Similar to LCC, the new method also takes advantage of the underlying geometry, and its complexity depends on the intrinsic dimensionality of the manifold instead of d. It has two main advantages over LCC. First, globally it can perfectly represent a quadratic function, which means that a smooth nonlinear function can be better approximated under the new scheme. Second, it requires a smaller number of anchor points than LCC, and thus reduces the computational cost.

The paper is organized as follows. In Section 2, we review the basic idea of LCC and the approximation bound that motivated the method. We then develop an improved bound by including quadratic approximation terms in Lemma 2.2. This bound is the theoretical basis of our new algorithm. Section 3 develops a more refined bound if the data lie on a manifold. We show in Lemma 3.1 that the new terms correspond to local tangent directions. Lemma 3.1 in Section 3 motivates the actual algorithm which we describe in Section 4. Section 5 shows the advantage of the improved LCC algorithm on some image classification problems. Concluding remarks are given in Section 6.

2. Local Coordinate Coding and its Extension

We are interested in learning a smooth nonlinear function $f(x)$ defined on a high dimensional space $\mathbb{R}^d$. In this paper, we denote by $\|\cdot\|$ an inner product norm on $\mathbb{R}^d$. The default choice is the Euclidean norm (2-norm):

$$\|x\| = \|x\|_2 = \sqrt{x_1^2 + \cdots + x_d^2}.$$

Definition 2.1 (Smoothness Conditions) *A function $f(x)$ on $\mathbb{R}^d$ is (α, β, ν) Lipschitz smooth with respect to a norm $\|\cdot\|$ if*

$$|\nabla f(x)^\top (x' - x)| \le \alpha \|x - x'\|,$$

and

$$\left| f(x') - f(x) - \nabla f(x)^\top (x' - x) \right| \le \beta \|x - x'\|^2,$$

and

$$\begin{aligned} &\left| f(x') - f(x) - 0.5(\nabla f(x') + \nabla f(x))^\top (x' - x) \right| \\ \le&\nu \|x - x'\|^3, \end{aligned}$$

where we assume $\alpha, \beta, \nu \ge 0$.

The parameter α is the Lipschitz constant of $f(x)$, which is finite if $f(x)$ is Lipschitz; in particular, if $f(x)$ is constant, then $\alpha = 0$. The parameter β is the Lipschitz derivative constant of $f(x)$, which is finite if the derivative $\nabla f(x)$ is Lipschitz; in particular, if $\nabla f(x)$ is constant (that is, $f(x)$ is a linear function of x), then $\beta = 0$. The parameter ν is the Lipschitz Hessian constant of $f(x)$, which is finite if the Hessian of $f(x)$ is Lipschitz; in particular, if the Hessian $\nabla^2 f(x)$ is constant (that is, $f(x)$ is a quadratic function of x), then $\nu = 0$. In other words, these parameters measure different levels of smoothness of $f(x)$: locally when $\|x - x'\|$ is small, α measures how well $f(x)$ can be approximated by a constant function, β measures how well $f(x)$ can be approximated by a linear function in x, and ν measures how well $f(x)$ can be approximated by a quadratic function in x. For local constant approximation, the error term $\alpha\|x - x'\|$ is the first order in $\|x - x'\|$; for local linear approximation, the error term $\beta\|x - x'\|^2$ is the second order in $\|x - x'\|$; for local quadratic approximation, the error term $\beta\|x - x'\|^3$ is the third order in $\|x - x'\|$. That is, if $f(x)$ is smooth with relatively small α, β, ν, the error term becomes smaller (locally when $\|x - x'\|$ is small) if we use a higher order approximation.

The following definition is copied from (Yu et al., 2009).

Definition 2.2 (Coordinate Coding) *A coordinate coding is a pair (γ, C), where $C \subset \mathbb{R}^d$ is a set of anchor points, and γ is a map of $x \in \mathbb{R}^d$ to $[\gamma_v(x)]_{v \in C} \in \mathbb{R}^{|C|}$ such that $\sum_v \gamma_v(x) = 1$. It induces the following physical approximation of x in $\mathbb{R}^d$:*

$$h_{\gamma,C}(x) = \sum_{v \in C} \gamma_v(x) v.$$

Moreover, for all $x \in \mathbb{R}^d$, we define the coding norm as

$$\|x\|_{\gamma,C} = \left(\sum_{v\in C} \gamma_v(x)^2\right)^{1/2} .$$

The importance of the coordinate coding concept is that if a coordinate coding is sufficiently localized, then a nonlinear function can be approximate by a linear function with respect to the coding. The following lemma is a slightly different version of a corresponding result in (Yu et al., 2009), where the definition of α was slightly different. We employs the current definition of α so that results in Lemma 2.1 and Lemma 2.2 are more compatible.

Lemma 2.1 (LCC Approximation) *Let (γ, C) be an arbitrary coordinate coding on $\mathbb{R}^d$. Let f be an (α, β, ν)-Lipschitz smooth function. We have for all $x \in \mathbb{R}^d$:*

$$\begin{aligned}&\left|f(x) - \sum_{v\in C}\gamma_v(x)f(v)\right|\\ \leq&\alpha\,\|x - h_{\gamma,C}(x)\| + \beta\sum_{v\in C}|\gamma_v(x)|\|v-x\|^2. \qquad (1)\end{aligned}$$

This result shows that a high dimensional nonlinear function can be globally approximated by a linear function with respect to the coding $[\gamma_v(x)]$, with unknown linear coefficients $[f(v)]_{v\in C}$. More precisely, it suggests the following learning method: for each x, we use its coding $[\gamma_v(x)] \in \mathbb{R}^{|C|}$ as features. We then learn a linear function of the form $\sum_v w_v\gamma_v(x)$ using a standard linear learning method such as SVM. Here $[w_v]$ is the unknown coefficient vector. The optimal coding can be learned using unlabeled data by optimizing the right hand side of (1) over unlabeled data. Details can be found in (Yu et al., 2009). The method is also related to sparse coding (Lee et al., 2007; Raina et al., 2007), which enforces sparsity but not locality. It was argued in (Yu et al., 2009) from both theoretical and empirical perspectives that locality is more important than sparsity. This paper follows the same line of theoretical consideration as in (Yu et al., 2009), and our theory relies on the locality concept as well.

A simple coding scheme is vector quantization, or VQ (Gray & Neuhoff, 1998), where $\gamma_v(x) = 1$ if $v = v_*(x)$ is the nearest neighbor of x in codebook C, and $\gamma_v(x) = 0$ otherwise. Since VQ is a special case of coordinate coding, its approximation quality can be characterized using Lemma 2.1 as follows. We have $h_{\gamma,C}(x) = v_*(x)$ and

$$\begin{aligned}&\left|f(x) - \sum_{v\in C}\gamma_v(x)f(v)\right|\\ \leq&\alpha\,\|x - v_*(x)\| + \beta\|v_*(x) - x\|^2.\end{aligned}$$

This method leads to local constant approximation of $f(x)$, where the main error is the first order term $\alpha\,\|x - v_*(x)\|$.

A better coding can be obtained by optimizing the right hand side of (1), which leads to the LCC method (Yu et al., 2009). The key advantage of LCC over VQ is that with appropriate local coordinate coding, $h_{\gamma,C}(x)$ linearly approximates x, hence the main error term $\|x - h_{\gamma,C}(x)\|$ can be significantly reduced. In particular, it was illustrated in (Yu et al., 2009) that for a smooth manifold, one can choose an appropriate codebook C with size depending on the intrinsic dimensionality such that the error term $\|x - h_{\gamma,C}(x)\| = O(\epsilon^2)$ is second order in ϵ, which represents the average distance of two near-by anchor points in C. In other words, the approximation power of LCC is local linear approximation. In contrast, the VQ method corresponds to locally constant approximation, where the error term $\|x - h_{\gamma,C}(x)\| = \|x - v_*(x)\| = O(\epsilon)$ is first order in ϵ. Therefore, from the function approximation point of view, the advantage of LCC over VQ is due to the benefit of 1st order (linear) approximation over 0th order (constant) approximation.

In the same spirit, we can generalize LCC by including higher order correction terms. One idea, which we introduce in this paper, is to employ additional directions into the coding, which can achieve second order approximation for relatively *locally flat* manifolds. The method is motivated from the following function approximation bound, which improves the LCC bound in Lemma 2.1.

Lemma 2.2 (Extended LCC Approximation) *Let (γ, C) be an arbitrary coordinate coding on $\mathbb{R}^d$. Let f be an (α, β, ν)-Lipschitz smooth function. We have for all $x \in \mathbb{R}^d$:*

$$\begin{aligned}&\left|f(x) - \sum_{v\in C}\gamma_v(x)\left(f(v) + 0.5\nabla f(v)^\top(x-v)\right)\right|\\ \leq&0.5\alpha\,\|x - h_{\gamma,C}(x)\| + \nu\sum_{v\in C}|\gamma_v(x)|\,\|v-x\|^3. \qquad (2)\end{aligned}$$

In order to use Lemma 2.2, we embed each x to extended local coordinate coding $[\gamma_v(x); \gamma_v(x)(x - v)]_{v\in C} \in \mathbb{R}^{|C|(1+d)}$. Now, a nonlinear function $f(x)$ can be approximated by a linear function of the extended coding scheme with unknown coefficients $\{f(v), 0.5\nabla f(v)\}$ (where $v \in C$). This method adds additional vector features $\gamma_v(x)(x-v)$ into the original coding scheme. Although the explicit number of features in (2) depends on the dimensionality d, we show later that for manifolds, the effective directions can be reduced to tangent directions that depend only on the intrinsic dimensionality of the underlying manifold.

If we compare (2) to (1), the first term on the right hand side is similar. That is, the extension does not improve this term. Note that this error term is small when x can be well approximated by a linear combination of local anchor points in C, which happens when the underlying manifold is relatively flat. The new extension improves the second term on the right hand side, where local linear approximation (measured by β) is replaced by local quadratic approximation (measured by ν). In particular, the second term vanishes if $f(x)$ is globally a quadratic function in x because $\nu = 0$. See discussions after Definition 2.1.

More generally, if $f(x)$ is a smooth function, then 2nd order approximation gives a 3rd order error term $O(\|v-x\|^3)$ in (2), compared to the 2nd order error term $O(\|v-x\|^2)$ in (1) resulted from 1st order approximation. The new method can thus yield improvement over the original LCC method if the second term on the right hand side of (1) is the dominant error term. In fact, our experiments show that this new method indeed improves LCC in practical problems. Another advantage of the new method is that the codebook size $|C|$ needed to achieve a certain accuracy becomes smaller, which reduces the computational cost for encoding: the encoding step requires solving a Lasso problem for each x, and the size of each Lasso problem is $|C|$.

Note that the extended coding scheme considered in Lemma 2.2 adds a d-dimensional feature vector $\gamma_v(x)(x-v)$ for each anchor $v \in C$. Therefore the complexity depends on d. However, if the data lie on a manifold, then one can reduce this complexity to the intrinsic dimensionality of the manifold using local tangent directions. We shall illustrate this idea more formally in the next section.

3. Data Manifolds

Similar to (Yu et al., 2009), we consider the following definition of manifold and its intrinsic dimensionality.

Definition 3.1 (Smooth manifold) *A subset $\mathcal{M} \subset \mathbb{R}^d$ is called a smooth manifold with intrinsic dimensionality $m = m(\mathcal{M})$ if there exists a constant $c(\mathcal{M})$ such that given any $x \in \mathcal{M}$, there exist m vectors (which we call tangent directions at x) $u_1(x), \ldots, u_m(x) \in \mathbb{R}^d$ so that $\forall x' \in \mathcal{M}$:*

$$\inf_{\gamma \in \mathbb{R}^m} \left\| x' - x - \sum_{j=1}^{m} \gamma_j u_j(x) \right\| \leq c(\mathcal{M}) \|x' - x\|^2.$$

Without loss of generality, we assume that the tangent directions $\|u_j(x)\| = 1$ for all x and j.

In this paper, we are mostly interested in the situation that the manifold is relatively locally flat, which means that the constant $c(\mathcal{M})$ is small. Algorithmically, the local tangent directions $u_k(v)$ can be found using local PCA, as described in the next section. Therefore for practical purpose one can always increase m to reduce the quantity $c(\mathcal{M})$. That is, we treat m as a tuning parameter in the algorithm. If m is sufficiently large, then $c(\mathcal{M})$ becomes small compared to β in Definition 2.1. If we set $m = d$, then $c(\mathcal{M}) = 0$. The approximation bound in the following lemma refines that of Lemma 2.2 because it only relies on local tangents with dimensionality m.

Lemma 3.1 (LCC with local Tangents) *Let $\mathcal{M}$ be a smooth manifold with intrinsic dimensionality $m = m(\mathcal{M})$. Then*

$$\begin{aligned}
&\Bigg| f(x) - \sum_{v \in C} f(v)\gamma_v(x) \\
&\quad -0.5 \sum_{v \in C} \sum_{k=1}^{m} (\nabla f(v)^\top u_k(v))((x-v)^\top u_k(v)) \Bigg| \\
&\leq 0.5\alpha \, \|x - h_{\gamma,C}(x)\| + 0.5\alpha c(\mathcal{M}) \sum_{v \in C} |\gamma_v(x)| \|x - v\|^2 \\
&\quad + \nu \sum_{v \in C} |\gamma_v(x)| \|x - v\|^3.
\end{aligned}$$

In this representation, we effectively use the reduced feature set $[\gamma_v(x); \gamma_v(x)(x-v)^\top u_k(v)]_{v \in C, k=1,\ldots,m}$, which corresponds to a linear dimension reduction of the extended LCC scheme in Lemma 2.2. These directions can be found through local PCA, as shown in the next section. The bound is comparable to Lemma 2.2 when $c(\mathcal{M})$ is small (with the appropriately chosen m), which is also assumed in Lemma 2.2 (see discussions thereafter). It improves the approximation result of the original LCC method in Lemma 2.2 if the main error term in (1) is the second term on the right hand side (again, this happens when $c(\mathcal{M})$ is small relatively to β).

While the result in Lemma 3.1 only justifies the new method we propose in this paper when $c(\mathcal{M})$ is small, we shall note that a similar argument holds when x lies on a noisy manifold. This is because in such case, the error caused by the first term on the right hand side of (1) has an inherent noise which cannot be reduced. Therefore it is more important to reduce the error caused by the second term on the right hand side of (1). A more rigorous statement can be developed in a style similar to Lemma 3.1, which we exclude from the current paper for simplicity.

4. Algorithm

Based on Lemma 3.1, we suggest the following algorithm which is a simple modification of the LCC method in (Yu et al., 2009) by including tangent directions that can be computed through local PCA.

- Learn LCC coding (γ, C) using the method described in (Yu et al., 2009).
- For each $v \in C$, using (local) PCA to find m principal components $u_1(v), \ldots, u_m(v)$ with weighted training data $\gamma_v(x)(x-v)$, where x belongs to the original training set.
- For each x, compute coding $\gamma_v(x)$ ($v \in C$), and form the extended coding $\tilde{\gamma}(x) = [s\gamma_v(x), \gamma_v(x)(x-v)^\top u_j(v)]_{v\in C, j=1,\ldots,m}$, where s is a positive scaling factor to balance the two types of codes.
- Learn a linear classifier of the form $w^\top \tilde{\gamma}(x)$, with $\tilde{\gamma}(x)$ as features.

In addition, we empirically find that standard sparse coding can be improved in a similar way, if we let (γ, C) in the first step be the result of sparse coding.

5. Experiments

In the following, we show that the improved LCC can achieve even better performance on image classification problems where LCC is known to be effective.

5.1. Handwritten Digit Recognition (MNIST)

Our first example is based on the MNIST handwritten digit recognition benchmark, where each data point is a 28×28 gray image, and pre-normalized into a unitary 784-dimensional vector. Our focus here is on checking whether a good nonlinear classifier can be obtained if we use LCC with local tangents as data representation, and then apply simple one-against-all linear SVMs.

In the experiments we try different sizes of bases. The parameters s, the weight of $\gamma_v(x)$, and m, the components of local PCA are both chosen based on cross-validation of classification results on the training data. It turns out that $s = 0$ and $m = 64$ is the best choice across different settings. The classification error rates are provided in Table 2.

In addition we compare the classification performances under different linear classifier on raw images, local kernel smoothing based on K-nearest neighbors, and linear classifiers using representations obtained from various unsupervised learning methods, including autoencoder based on deep belief networks (DBN) (Hinton & Salakhutdinov, 2006), Laplacian eigenmaps (LE) (Belkin & Niyogi, 2003), locally linear embedding (LLE) (Roweis & Saul, 2000), VQ coding based on K-means, sparse coding (SC), and original LCC. We note that, like most of other manifold learning approaches, LE or LLE is a transductive method which has to incorporate both training and testing data in training.

The comparison results are summarized in Table 1. Both SC and LCC perform quite good for this nonlinear classification task, significantly outperforming linear classifiers on raw images. In addition, LCC using local tangents is consistently better than all the other methods across various basis sizes. Among those compared methods in Table 1, we note that the error rate 1.2% of DBN reported in (Hinton & Salakhutdinov, 2006) was obtained via unsupervised pre-training followed by supervised backpropagation. The error rate based on unsupervised training of DBN is about 1.90%. Therefore our result is the *state-of-the-art* among those that are based on unsupervised feature learning on MNIST, without using any convolution operation. The results also suggest that, compared with original LCC using 4096 bases, the improved version can achieve a similar accuracy by using only 512 bases.

Table 1. Error rates (%) of MNIST classification with different methods.

Methods	Error Rate
Linear SVM with raw images	12.0
Linear SVM with VQ	3.98
Local kernel smoothing	3.48
Linear SVM with LE	2.73
Linear SVM with LLE	2.38
Linear classifier with DBN	1.90
Linear SVM with SC	2.02
Linear SVM with LCC	1.90
Linear SVM with improved LCC	**1.64**

Table 2. Error rates (%) of MNIST classification with different basis sizes, by using linear SVM.

|C|	512	1024	2048	4096
LCC	2.64	2.44	2.08	1.90
Improved LCC	**1.95**	**1.82**	**1.78**	**1.64**

5.2. Image Classification (CIFAR10)

The CIFAR-10 dataset is a labeled subset of the 80 million tiny images dataset (Torralba et al., 2008). It was collected by Vinod Nair and Geoffrey Hinton (Krizhevsky & Hinton, 2009), where all the images were manually labeled. The dataset consists of 60000 32×32 color images in 10 classes, 6000 images per class. There are 50000 training images and 10000 test images. The dataset is divided into five training batches and one test batch, each with 10000 images. The test batch contains exactly 1000 randomly-selected images from each class. The training batches contain the remaining images in random order, but some training batches may contain more images from one class than another. Between them, the training batches contain exactly 5000 images from each class. Example images are shown in Figure 1.

We treat each color image as a $32 \times 32 \times 3 = 3072$ dimensional vector, and pre-normalize it to ensure the unitary length of each vector. Due to the high level of redundancy cross R/G/B channels, we reduce the dimensionality to 512 by using PCA, while still retaining 99% of the data variances. Since our purpose here is to obtain good feature vectors for linear classifiers, our baseline is a linear SVM directly trained on this 512-dimensional feature representation. We train LCC with different dictionary sizes on this data set and then apply both LCC coding and the improved version with local tangents. Linear SVMs are then trained on the new presentations of the training data. The classification accuracy of both LCC methods under different dictionary sizes is given in Table 4. Similar to what we did for MNIST, the optimal parameters $s = 10$ and $m = 256$ are determined via cross-validation on training data. We can see that local tangent expansion again consistently improves the quality of features in terms of better classification accuracy. It is also observed that a larger dictionary size leads to a better classification accuracy, as the best result is obtained with the dictionary size 4096. The trend implies a better performance might be reached if we further increase the dictionary size, which however requires more computation and unlabeled training data.

The prior state of the art performance on this data set was obtained by Restricted Boltzmann Machines (RBMs) reported in (Krizhevsky & Hinton, 2009), whose results are listed in Table 3. The compared methods are

- 10000 Backprop autoencoder: the features were learned from the 10000 logistic hidden units of a two-layer autoencoder neural network trained by back propagation.
- 10000 RBM Layer2: a stack of two RBMs with two layers of hidden units, trained with contrast divergence.
- 10000 RBM Layer2 + finetuning: the feedforward weights of RBMs are fine-tuned by supervised back propagation using the information labels.
- 10000 RBM: a layer of RBM with 10000 hidden units, which produces 10000 dimensional features via unsupervised contrastive divergence training.
- 10000 RBM + finetuning: the single layer RBM is further trained by supervised back propagation. This method gives the best results in the paper.

As we can see, both results of LCC significantly outperform the best result of RBMs, which suggests that the feature representations obtained by LCC methods are very useful for image classification tasks.

Table 3. Classification accuracy (%) on CIFAR-10 image set with different methods.

Methods	Accuracy Rate
Raw pixels	43.2
10000 Backprop autoencoder	51.5
10000 RBM Layer2	58.0
10000 RBM Layer2 + finetuning	62.2
10000 RBM	63.8
10000 RBM + finetuning	64.8
Linear SVM with LCC	72.3
Linear SVM with improved LCC	**74.5**

Table 4. Classification accuracy (%) on CIFAR-10 image set with different basis sizes, by using linear SVM.

|C|	512	1024	2048	4096
LCC	50.8	56.8	64.4	72.3
Improved LCC	**55.3**	**59.7**	**66.8**	**74.5**

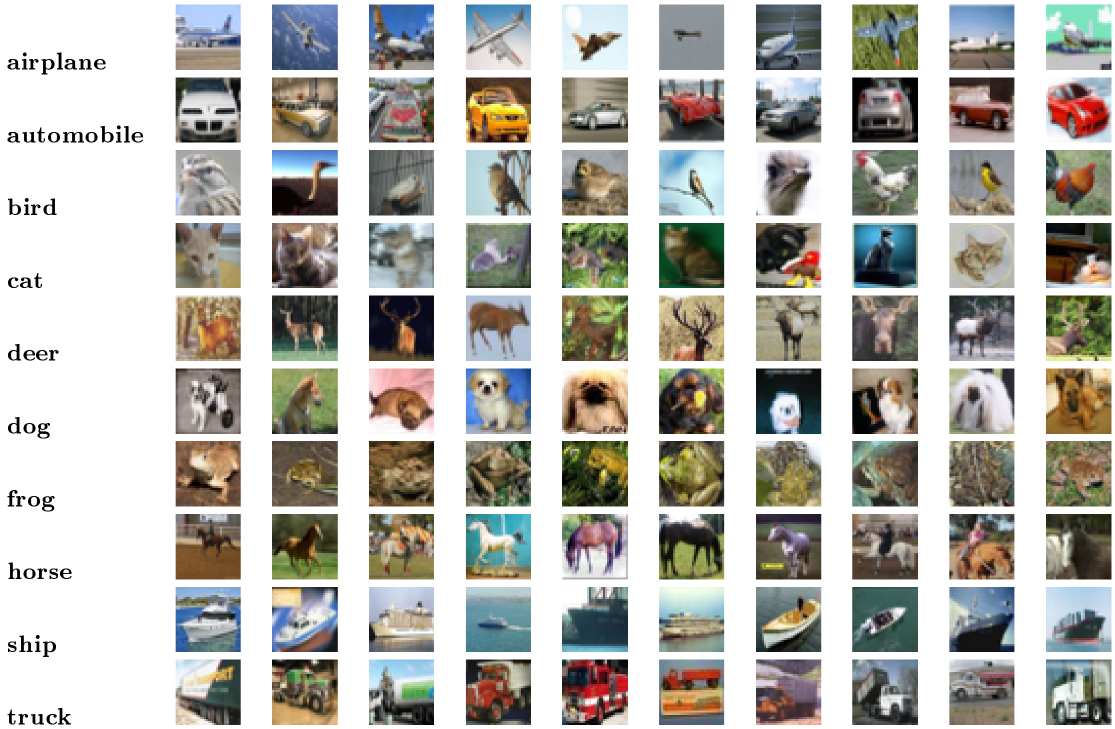

Figure 1. Examples of tiny images from CIFAR-10

6. Discussions

This paper extends the LCC method by including local tangent directions. Similar to LCC, which may be regarded as the soft version of VQ that linearly interpolates local VQ points, the new method may be regarded as the soft version of local PCA that linearly interpolates local PCA directions. This soft interpolation allows the possibility to achieve second order approximation when the underlying data manifold is relatively locally flat, as shown in Lemma 2.2 and Lemma 3.1.

Experiments demonstrate that this new method is superior to LCC for image classification. First, the new method requires a significantly smaller number of anchor points to achieve a certain level of accuracy, which is important computationally because the coding step is significantly accelerated. Second, it improves prediction performance on some real problems.

However, theoretically, the bound in Lemma 3.1 only shows improvement over the LCC bound in Lemma 2.1 when the underlying manifold is locally flat (although similar conclusion holds when the manifold is noisy, as remarked after Lemma 3.1). At least theoretically our analysis does not show how much value the added local tangents has over LCC when the underlying manifold is far from locally flat. Since we do not have a reliably way to empirically estimate the local flatness of a data manifold (e.g. the quantity $c(\mathcal{M})$ in Definition 3.1), we do not have good empirical results illustrating the impact of manifold's "flatness" either. Therefore it remains an open issue to develop other coding schemes that are provably better than LCC even when the underlying manifold is not locally flat.

In our experiments, we treat each image as a single data vector for coding. But in the practice of image classification, to handle spatial invariance, we need to apply coding methods on local patches of the image and then use some pooling strategy on top of that. This is well-aligned with the architecture of convolution neural networks (LeCun et al., 1998). However, what is the best strategy for pooling has not been understood theoretically. In particular, we want to understand the interplay of coding on local patches and the classification function defined on images, which remains an interesting open problem.

References

Belkin, Mikhail and Niyogi, Partha. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput.*, 15(6):1373–1396, 2003. ISSN 0899-7667.

Everingham, Mark. Overview and results of the classification challenge. *The PASCAL Visual Object Classes Challenge Workshop at ICCV*, 2009.

Gray, Robert M. and Neuhoff, David L. Quantization. *IEEE Transaction on Information Theory*, pp. 2325 – 2383, 1998.

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

Krizhevsky, A. and Hinton, G. E. Learning multiple layers of features from tiny images. Technical report, Computer Science Department, University of Toronto, 2009.

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

Lee, Honglak, Battle, Alexis, Raina, Rajat, and Ng, Andrew Y. Efficient sparse coding algorithms. *Neural Information Processing Systems (NIPS)*, 2007.

Raina, Rajat, Battle, Alexis, Lee, Honglak, Packer, Benjamin, and Ng, Andrew Y. Self-taught learning: Transfer learning from unlabeled data. *International Conference on Machine Learning*, 2007.

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

Torralba, A., Fergus, R., and Freeman, W.T. 80 million tiny images: a large dataset for non-parametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11): 1958–1970, 2008.

Yu, Kai, Zhang, Tong, and Gong, Yihong. Nonlinear learning using local coordinate coding. In *NIPS' 09*, 2009.

A. Proofs

For notation simplicity, let $\gamma_v = \gamma_v(x)$ and $x' = h_{\gamma,C}(x) = \sum_{v \in C} \gamma_v v$.

A.1. Proof of Lemma 2.1

We have

$$\begin{aligned}
&\left| f(x) - \sum_{v \in C} \gamma_v f(v) \right| \\
=&\left| \sum_{v \in C} \gamma_v \Big(f(v) - f(x) - \nabla f(x)^\top (v - x') \Big) \right| \\
\leq&\left| \nabla f(x)^\top (x - x') \right| + \left| \sum_{v \in C} \gamma_v \Big(f(v) - f(x) - \nabla f(x)^\top (v - x) \Big) \right| \\
\leq&\left| \nabla f(x)^\top (x - x') \right| + \beta \sum_{v \in C} |\gamma_v| \|x - v\|^2 \\
\leq&\alpha \|x - x'\| + \beta \sum_{v \in C} |\gamma_v| \|x - v\|^2 .
\end{aligned}$$

A.2. Proof of Lemma 2.2

We have

$$\begin{aligned}
&\left| f(x) - \sum_{v \in C} \gamma_v \Big(f(v) + 0.5 \nabla f(v)^\top (x - v) \Big) \right| \\
=&\left| \sum_{v \in C} \gamma_v \Big(f(v) - f(x) - 0.5 \nabla f(x)^\top (v - x') \right. \\
&\quad \left. + 0.5 \nabla f(v)^\top (x - v) \Big) \right| \\
\leq&0.5 \left| \nabla f(x)^\top (x - x') \right| \\
&\quad + \left| \sum_{v \in C} \gamma_v \Big(f(v) - f(x) - 0.5 (\nabla f(x) + \nabla f(v))^\top (v - x) \Big) \right| \\
\leq&0.5 \left| \nabla f(x)^\top (x - x') \right| + \nu \sum_{v \in C} |\gamma_v| \|x - v\|^3 \\
\leq&0.5 \alpha \|x - x'\| + \nu \sum_{v \in C} |\gamma_v| \|x - v\|^3 .
\end{aligned}$$

A.3. Proof of Lemma 3.1

Let P_v be the projection operator from $\mathbb{R}^d$ to the subspace spanned by $u_1(v), \ldots, u_m(v)$ with respect to the inner product norm $\|\cdot\|$. We have

$$\begin{aligned}
&\left| f(x) - \sum_{v \in C} \gamma_v \Big(f(v) + 0.5 \nabla f(v)^\top P_v (x - v) \Big) \right| \\
\leq&\left| f(x) - \sum_{v \in C} \gamma_v \Big(f(v) + 0.5 \nabla f(v)^\top (x - v) \Big) \right| \\
&\quad + \left| 0.5 \sum_{v \in C} \gamma_v \nabla f(v)^\top (I - P_v)(x - v) \right| \\
\leq&0.5 \alpha \|x - x'\| + \nu \sum_{v \in C} |\gamma_v| \|x - v\|^3 \\
&\quad + 0.5 \alpha \sum_{v \in C} |\gamma_v| \, \|(I - P_v)(x - v)\| .
\end{aligned}$$

Now Definition 3.1 implies that $\|(I - P_v)(x - v)\| \leq c(\mathcal{M}) \|x - v\|^2$. We thus obtain the desired bound.