Semi-Supervised Learning with Sparse Distributed Representations

David Ziegler
dziegler@stanford.edu
CS 229 Final Project

1 Introduction

For many machine learning applications, labeled data may be very difficult or costly to obtain. For instance in the case of speech analysis, the average annotation time for a one hour telephone conversation transcript is 400 hours.[7] To circumvent this problem, one can use semi-supervised learning algorithms which utilize unlabeled data to improve performance on a supervised learning task. Since unlabeled data is typically much easier to obtain, this can be an attractive approach.

In this paper, we combine semi-supervised learning with two types of sparse distributed representations: local and global. In a local sparse distributed representation we attempt to find local sparse features, while in a global sparse distributed representation we attempt to represent our test set using a sparse combination of the training set. In both cases, sparsity turns out to have desirable properties which we can leverage in semi-supervised learning.

2 Local Sparse Distributed Representations

Sparse coding is very well suited toward the extraction local sparse features. The goal of sparse coding is to represent input vectors as a sparse approximate weighted linear combination of basis vectors. That is, for input vector $y \in \mathbb{R}^k$,

$$y \approx \sum_j b_j s_j = Bs \tag{1}$$

where $b_1, \cdots, b_n \in \mathbb{R}^k$ and $s \in \mathbb{R}^n$ is a sparse vector of coefficients. Unlike similar methods such as PCA, the basis set B founded in sparse coding can be overcomplete ($n > k$), and can represent nonlinear features of x. To find the optimal B and s we solve the following optimization problem as formulated by [3]:

$$\text{minimize}_{b,a} \quad \sum_i ||x^{(i)} - \sum_j a_j^{(i)} b_j||_2^2 + \beta ||a^{(i)}||_1 \tag{2}$$

$$\text{s.t} \quad ||b_j||_2 \le 1,\ \forall j \in 1, \cdots, s \tag{3}$$

3 The Structural Self-Taught Learning Algorithm

Traditionally, semi-supervised algorithms have assumed that the unlabeled data and the labeled data are drawn from the same distribution. A recent semi-supervised framework proposed by Raina et al[4] called self-taught learning abandons this assumption, and only requires that the unlabeled data be within the same problem domain, or modality. The approach taken by the authors in [4] is to learn a set of bases on the unlabeled data via sparse coding, and then represent the labeled data as a sparse linear combination of the learned bases. This allows them to map their labeled data into a higher level represenation, which they can then perform supervised learning on. The algorithm is summarized below. This algorithm falls under the more general case of self-taught learning

Algorithm 1 Self-taught Learning via Sparse Coding

```
1: Input: Labeled training set $T = (x_l^{(1)}, y_l^{(1)}), \cdots, (x_l^{(m)}, y_l^{(m)})$. Unlabeled data $x_u^{(1)}, \cdots, x_u^{(k)}$.
2: Using unlabeled data $x_u^{(i)}$, solve the optimization problem (1) to obtain bases $B$. Compute features for the classification task to obtain a new labeled training set $\hat{T} = \{(\hat{a}(x_l^{(i)}), y^{(i)})\}_{i=1}^m$, where $\hat{a}(x_l^{(i)}, y^{(i)}) = \text{argmin}_{a^{(i)}} ||x_l^{(i)} - sum_j a_j^{(i)} b_j||_2^2 + \beta||a^{(i)}||_1$. Learn a classifier $C$ by applying a supervised learning algorithm to the labeled training set $\hat{T}$.
3: Ouput: Learned classifier for the classification task.
```

algorithms which we call structural self-taught learning. In structural self-taught learning, we attempt to extract an abstract underlying structure from the unlabeled data, use this underlying structure to project our labeled data into a higher level represenation, and then perform supervised learning on the labeled data. This approach to self-taught learning opens up two important questions which we will attempt to address:

1. How does the choice of our structural mapping function affect performance on our supervised learning task (in this case, sparse coding)?

2. How does the similarity between the labeled and unlabeled data affect performance on our supervised learning task?

While there has been good theoretical work done recently in semi-supervised learning[1] and transfer learning[2], none of these approaches are applicable to the case of self-taught learning, where we do not assume anything about the distribution of the unlabeled data.

We define the compatibility function $\chi : D_u \times D_l \times f \to [0,1]$ relating the compatability between unlabeled distribution D_u, labeled distribution D_l, and structural mapping function f, as $\chi(D_u, D_l, f) = 1 - E_{x_l \in D_l}[\chi(x_u, x_l, f)]$. In the case of sparse coding,

$$\chi(x_u, x_l, a) = 1 - E[x_l - B\hat{a}(x_l)] \tag{4}$$

Essentially, we measure the compatibility between an unlabeled distribution, a labeled distribution, and a mapping function, by the reconstruction error with respect to the the optimal basis found over the unlabeled data.

4 MNIST Experiments

For our experiments with self-taught learning, we used the MNIST handwritten digits dataset for labeled data. Unlabeled data was taken from three sources: natural images, computer font characters, and MNIST. The number of bases was kept fixed at 512 and all images were resized to 14×14 pixels. We used two classifiers: a linear and gaussian svm and compared the performance between the raw pixel intensities and the sparse codes.

The results generally support our hypothesis that a higher compatibility function yields better results on the sparse coding classification relative to the raw pixel intensities (figure 1). The results are more pronounced for the linear classifier. This could be that the gaussian kernel is better suited towards raw pixel intensities, and we note that to allow for a fair comparison between the raw and sparse coding features, we did not use the sparse coding kernel derived in [4].

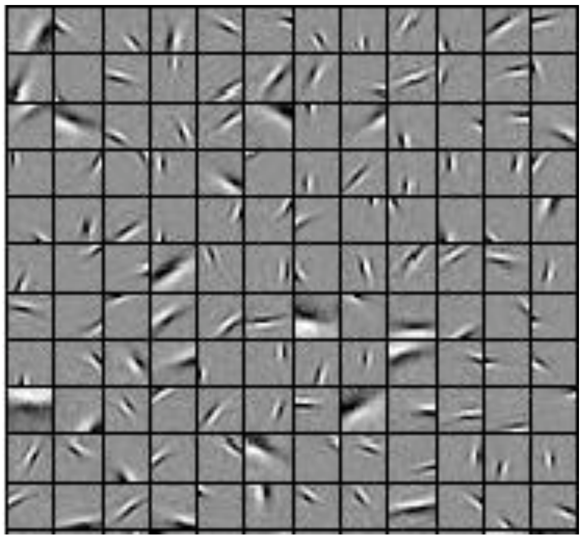


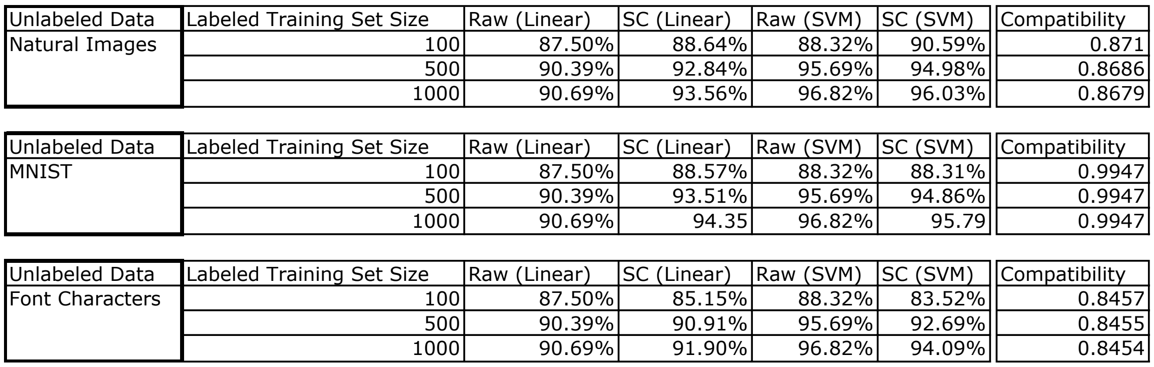
Figure 2: Sample bases learned from natural images

5 Global Sparse Distributed Representations

A different approach which we call a global sparse distributed representation is to represent features from a test example using a sparse combination of features from the training set. Unlike local sparse distributed representations, it turns out that within the context of a global sparse distributed representation, the selection of features becomes completely irrelevant. Recent research has shown that for face recognition, if the representation is sparse enough, random linear projected features perform just as well or better than feature selection for via Eigenfaces, Laplacianfaces, or Fisherfaces [6]. Building off of the pattern recognition algorithm used in [6] we devise a simple semi-supervised learning algorithm which utilizes a bootstrapping approach to expand our training set.

We assume that images within the same category lie on a low-dimensional linear subspace. The k individual subspaces can be represented by matrices $A_1, A_2, \cdots, A_k$ where each column in A_i is a training sample from class i. Let y be an image from the test set, and $A = [A_1 A_2 \cdots A_k]$. Then ideally,

$$y = Ax_0 \in \mathbb{R}^m \tag{5}$$



Unlabeled Data	Labeled Training Set Size	Raw (Linear)	SC (Linear)	Raw (SVM)	SC (SVM)	Compatibility
Natural Images	100	87.50%	88.64%	88.32%	90.59%	0.871
	500	90.39%	92.84%	95.69%	94.98%	0.8686
	1000	90.69%	93.56%	96.82%	96.03%	0.8679

Unlabeled Data	Labeled Training Set Size	Raw (Linear)	SC (Linear)	Raw (SVM)	SC (SVM)	Compatibility
MNIST	100	87.50%	88.57%	88.32%	88.31%	0.9947
	500	90.39%	93.51%	95.69%	94.86%	0.9947
	1000	90.69%	94.35	96.82%	95.79	0.9947

Unlabeled Data	Labeled Training Set Size	Raw (Linear)	SC (Linear)	Raw (SVM)	SC (SVM)	Compatibility
Font Characters	100	87.50%	85.15%	88.32%	83.52%	0.8457
	500	90.39%	90.91%	95.69%	92.69%	0.8455
	1000	90.69%	91.90%	96.82%	94.09%	0.8454

Figure 1: MNIST Results

where $x_0 = [0, \cdots, 0, \alpha_{i,1}, \alpha_{i,2}, \cdots, \alpha_{i,n_i}, 0, \cdots, 0,]^T \in \mathbb{R}^n$ is a coefficient vector whose entries are mostly zero except those associated with category i. Thus, a new test image should be predominantly represented using training images from the same subject, and this representation will be naturally sparse if the number of categories k is reasonably large.

Let $R \in \mathbb{R}^{dxm}$ be a transform matrix with $d \ll m$. Multiplying both sides of (5) by R yields $\tilde{y} = Ry = RAx_0 = \tilde{A}x_0$. Recent developments in compressed sensing have shown that the desired x_0 is the solution to the l^1 minimization problem:

$$\min ||x||_1 \quad \text{s.t. } \tilde{y} = \tilde{A}x \tag{6}$$

Furthermore, if x has $p \ll n$ nonzeros, then

$$d \geq 2p \log(n/d) \tag{7}$$

random measurements are sufficient for sparse recovery with high probability[6]. Thus even with randomly selected facial features, if the dimension of the image is sufficiently large then we should still be able to classify the test image.

Let $\delta_i(x) \in \mathbb{R}^n$ be a vector whose only nonzero entries are the entries in x associated with category i. The classification algorithm is then given by Algorithm 2 [6]. Note that the minimization problem in step 3 is slightly different than in (6) to model noise and error in the data.

6 Semi-Supervised Learning in Global Sparse Distributed Representations

We define the Sparsity Concentration Index as in [6]:

$$\text{SCI}(x) = \frac{k}{k-1} \cdot \max_i ||\delta_i(x)||_1/||x||_1 - 1 \in [0,1] \tag{8}$$

The SCI provides a measure of how concentrated the coefficients are on a specific category. When SCI = 1, the test image is represented using only images from a single subject, and when SCI = 0, the coefficients are spread evenly over all categories. We also define $\hat{r}$ to be the ratio between the smallest and second smallest residual.

The residuals calculated in Algorithm 1 measures how well the the sparse representation approximates the test image, while the SCI and $\hat{r}$ give us a measure of how good the representation is in terms of localization.

Algorithm 2 Recognition via Sparse Representation

```
1: Input: a matrix of training images $A \in \mathbb{R}^{m \times n}$ for
   $k$ subjects, a linear feature transform $R \in \mathbb{R}^{d \times m}$,
   a test image $y \in \mathbb{R}^m$, and an error tolerance $\epsilon$.
2: Compute features $\tilde{y} = Ry$ and $\tilde{A} = RA$ and normalize $\tilde{y}$ and columns of $\tilde{A}$ to unit length.
3: Solve: min $||x||_1$   s.t. $||\tilde{y} - \tilde{A}x||_2 \leq \epsilon$
4: Compute residuals $r_i(y) = ||\tilde{y} - \tilde{A}\delta_i(x)||_2$ for $i = 1, \cdots, k$
5: Output: identity$(y)$ = argmin$_i r_i(y)$
```

After running Algorithm 3 on our unlabeled data, we then run Algorithm 2 to perform the supervised learning task. Alternatively, we can run Algorithm 3 on the test set, augmenting our training set with images

Algorithm 3 Bootstrapping a Global Sparse Representation Classifier

```
1: Input: a matrix of training images $A \in \mathbb{R}^{m\times n}$ for
   $k$ subjects, a linear feature transform $R \in \mathbb{R}^{d\times m}$,
   a matrix of unlabeled images $Y \in \mathbb{R}^{m\times q}$, and an
   error tolerance $\epsilon$.
2: loop
3:     $numLabeled = 0$
4:     for all $y \in Y$ do
5:         Compute features $\tilde{y} = Ry$ and $\tilde{A} = RA$
           and normalize $\tilde{y}$ and columns of $\tilde{A}$ to
           unit length
6:         Solve: $\min ||x||_1 \quad \text{s.t. } ||\tilde{y} - \tilde{A}x||_2 \leq \epsilon$
7:         Compute residuals $r_i(y) = ||\tilde{y} - \tilde{A}\delta_i(x)||_2$
           for $i = 1, \cdots, k$
8:         if SCI$(x) \geq \tau$ and $\hat{r} \geq \sigma$ then
9:             $j = \text{identity}(y) = \text{argmin}_i r_i(y)$
10:            Append $y$ to $A_j$ and remove $y$ from
               $Y$
11:            Update: $A = [A_1 A_2 \cdots A_k]$
12:            $numLabeled+ = 1$
13:        end if
14:    end for
15:    if $numLabeled = 0$ then
16:        break
17:    end if
18: end loop
19: Output: Augmented training matrix $A \in \mathbb{R}^{m\times p}$,
    $p \geq n$
```

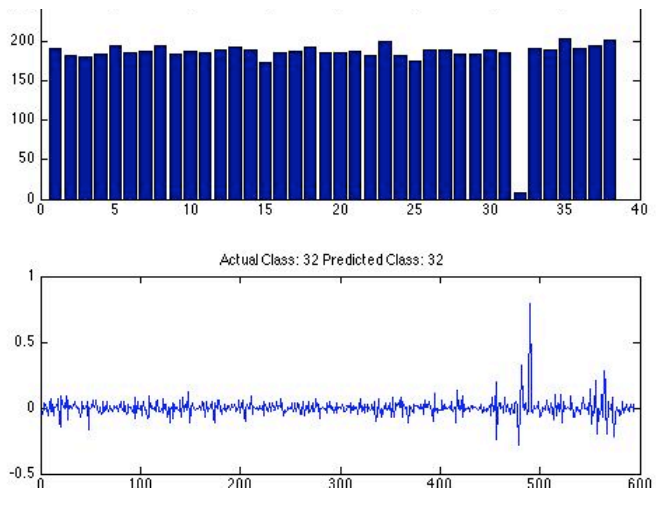


Figure 3: Top:Residuals, Bottom: x obtained from (6)

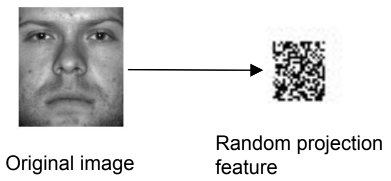


Figure 4: Random Feature Example

from the test set, and then run Algorithm 2 on the remaining test images.

This algorithm has two potential drawbacks. The first is that if τ and σ are set improperly, then we might add misclassified images to our training set, which would then cause compounding errors in later classifications, thus obscuring the categories in our training set. In our experiments this problem did not arise, but this is an inherent weakness in any bootstrapping algorithm. However, so long as the conditions in (7) are satisfied, the probability of a sparse incorrect reconstruction occurring has very low probability. Thus by setting τ and σ properly, the probability of a misclassified images being added to the training set is very low.

The other concern is raised by the theory underlying (7), which is that by increasing our training set while keeping the dimensionality and number of categories fixed, we reduce the probability that we can successfully reconstruct the test image since we are potentially decreasing the sparsity of x. For category i with n_i examples, and $R \in \mathbb{R}^{m\times n}$ if

$$n_i + |support(\epsilon)| > m/3 \tag{9}$$

then we should not expect to perfectly recover x and ϵ [5]. However, this is generally not a problem since so long as the dimension of R is large enough. If one suspects that this might be a problem, one could stop the algorithm once the sparsity started getting close to the bound in (4).

7 Experiments with Face Recognition

Experiments were conducted the Extended Yale B database. The database consists of 2,414 cropped and normalized frontal-face images of 38 individuals under various laboratory controlled lighting conditions. The feature space dimension chosen was 504.

We first randomly split half the data into a training

	Original	Bootstrapped	Original	Bootstrapped
Training Size	103	606	594	1003
Dimension	504	504	504	504
Accuracy	62%	72%	84%	90%

Figure 5: Training set augmented with unlabeled set

	Original	Bootstrapped
Training Size	1207	2133
Dimension	504	504
Accuracy	94.65%	96.51%

Figure 6: Training set augmented with test set

set and half into a test set. Then a portion of the training set was randomly split off and randomly permuted into a third "unlabeled training set". We then ran our bootstrapping algorithm on this unlabeled set to augment our training set, and then tested our augmented training set with the separate test set (figure 5). The algorithm does demonstrate significant improvement on the test set, although the amount it improves performance seems to decrease asymptotically as the initial training set size is increased.

We also ran a third experiment where we used the entire training set and ran the bootstrapping algorithm on the test set, augmenting the training set with the test set. We again get a performance improvement, although the effect is small since we start out with a large training set (figure 6). In all of the bootstrapping experiments, τ and σ were set conservatively to ensure that no unlabeled data was added to the training set and misclassified. In all the above experiments, post-experiment analysis showed that the unlabeled data which was added to the training set was all correctly classified.

8 Future Work and Conclusions

We provided a way which allows one to quantitatively determine how the compatibility between the unlabeled and labeled data will affect performance on the supervised learning task for a given structural mapping function. To make this result more robust, it would be nice to show this compatibility with non-image data, and to experiment with different mapping functions such as PCA. Also, it would be interesting to see at what point of incompatibility the unlabeled data stopped helping performance on the supervised task.

The global sparse distributed representation is a fairly recent development, with many attractive properties. In particular, the ability to classify using random features. The bootstrapping algorithm shown here is fairly simplistic but surprisingly robust and helpful in boosting performance. To explore this algorithm further, it would be useful to perform experiments where the parameters were set too low, causing misclassifications to enter our training set, and see how performance degrades as a result.

9 Acknowledgements

Thanks to Honglak Lee for helping me with my sparse coding implementation.

References

[1] M. Balcan and A. Blum. A pac-style model for learning from labeled and unlabeled data, 2005.

[2] J. Baxter. Theoretical models of learning to learn, 1997.

[3] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y. Ng. Efficient sparse coding algorithms. In *Neural Information Processing Systems*, 2006.

[4] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y. Ng. Self-taught learning: transfer learning from unlabeled data. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 759–766, New York, NY, USA, 2007. ACM.

[5] John Wright, Arvind Ganesh, Allen Yang, and Yi Ma. Robust face recognition via sparse representation, 2007.

[6] Allen Y. Yang, John Wright, Yi Ma, and S. Shankar Sastry. Feature selection in face recognition: A sparse representation perspective. Technical Report UCB/EECS-2007-99, EECS Department, University of California, Berkeley, Aug 2007.

[7] X. Zhu. Semi-supervised learning literature survey. Technical Report TR 1530, Computer Sciences Department, University of Wisconsin - Madison, June 2007.