Exponential family sparse coding with application to self-taught learning with text documents

Honglak Lee HLLEE@CS.STANFORD.EDU
Rajat Raina RAJATR@CS.STANFORD.EDU
Alex Teichman TEICHMAN@CS.STANFORD.EDU
Andrew Y. Ng ANG@CS.STANFORD.EDU
Stanford University, Stanford, CA 94305 USA

Abstract

Sparse coding is an unsupervised learning algorithm for finding concise, slightly higher-level representations of an input, and has been successfully applied to self-taught learning (Raina et al., 2007), where the goal is to use unlabeled data to help on a supervised learning task, even if the unlabeled data cannot be associated with the labels of the supervised task. However, sparse coding uses a Gaussian noise model and a quadratic loss function, and thus performs poorly if applied to binary valued, integer valued, or other non-Gaussian data, such as text. Drawing on ideas from Generalized linear models (GLMs), we present a generalization of sparse coding to learning with data drawn from any exponential family distribution (such as Bernoulli, Poisson, etc). This gives a method that we argue is much better suited to model other data types than Gaussian. We present an efficient algorithm for solving the optimization problem defined by this model. We also show that the new model results in significantly improved self-taught learning performance when applied to text data.

1. Introduction

We consider the "self-taught learning" problem, in which we are given just a few labeled examples for a classification task, and also large amounts of unlabeled data that is only mildly related to the task (Raina et al., 2007; Weston et al., 2006). Specifically, the unlabeled data may not share the class labels or arise from the same distribution. For example, given only a few labeled examples of webpages about "Baseball" or "Football", along with access to a large corpus of unrelated webpages, we might want to accurately classify new baseball/football webpages. The ability to use such easily available unlabeled data has the potential to greatly reduce the effect of data sparsity, and thus greatly improve performance on labeling or tagging applications in language processing.

Our approach uses the unlabeled data to learn a high-level representation of the inputs, and then using this representation to provide features for classification. Raina et al. demonstrate such a method for domains such as image classification, using the "sparse coding" model (Olshausen & Field, 1996). In this model, given real-valued vectors $x \in \mathbb{R}^k$ as inputs, we attempt to learn a large set of basis vectors $b_1, b_2, \ldots, b_n \in \mathbb{R}^k$ such that the input can be represented as a linear combination of only a few basis vectors: i.e., $x \approx \sum_j b_j s_j$, where s_j is the weight (or "activation") for basis vector b_j, and most s_j values are zero (or, the vector s is *sparse*). Informally, the activation vector s used to reconstruct an input x often captures the few "most important" patterns in x. For example, when this model is applied to images, the basis vectors learnt by the model represent various edge patterns, and thus s captures the edges present in an image input x. Indeed, when the activations s are used as features in a standard supervised learning algorithm (such as an SVM), the generalization performance is often better than when using the raw input x as features.

The sparse coding algorithm for learning basis vectors is based on a Gaussian noise model for input x: $P(x|b,s) = \mathcal{N}(\sum_j b_j s_j, \sigma^2 I)$, where σ^2 is fixed. A sparse prior $P(s) \propto \prod_j \exp(-\beta|s_j|)$ is assumed on the activations to penalize nonzero activations. Then, given unlabeled examples $\{x^{(1)}, x^{(2)}, \ldots\}$ and the corresponding activations $\{s^{(1)}, s^{(2)}, \ldots\}$, good basis vectors are estimated by solving the MAP optimization problem, which is equivalent to solving:[1]

$$\min_{\{b_j\},\{s^{(i)}\}} \quad \tfrac{1}{2\sigma^2} \textstyle\sum_i \|x^{(i)} - \sum_{j=1}^n b_j s_j^{(i)}\|^2 + \beta \sum_{i,j} |s_j^{(i)}|$$
$$\text{subject to} \qquad \|b_j\|^2 \le c,\ \forall j = 1, ..., n.$$

[1] Following previous work, we constrain the norm of each basis vector b_j to make the problem well-posed.

This optimization problem is convex separately in the b and s variables (though not jointly convex). The problem can be solved efficiently by alternating minimization over b and s variables (Lee et al., 2007). Finally, given learnt basis vectors b, the features for a new input example x are derived by estimating: $\arg\max_s P(s|x,b) = \arg\max_s P(x|b,s)P(s)$.

2. Self-taught learning for text data

The probabilistic model used in sparse coding assumes that the input vectors x are real-valued, and that they can be well described using a Gaussian noise model. However, this is an inappropriate assumption for text data, which is often represented as a binary "bag-of-words" vector $x \in \{0,1\}^k$, where the i-th feature is 1 if the i-th word in our vocabulary occured in the document, or as a word-counts vector $x \in \{0,1,2,\dots\}^k$, where the i-th feature represents the number of times the i-th word occured in the document. In either case, the input vectors are very poorly modeled by a continuous Gaussian distribution (which could take fractional, or negative values). It is thus hardly surprising that when sparse coding is applied to a self-taught learning task for text, it only leads to small gains in accuracy, even with huge amounts of unlabeled data; in some cases, it can even hurt performance.

To address this problem, in this paper we generalize the Gaussian probabilistic model behind sparse coding in a principled way to "exponential family" of distributions. This class of distributions is large enough to include most commonly used distributions (including the Gaussian, Bernoulli and Poisson distributions among others), but also guarantees crucial properties that make efficient learning and inference possible (e.g., the maximum likelihood learning problem is convex for any distribution in the family) (McCullagh & Nelder, 1989). We call our model *exponential family sparse coding*, and to differentiate it from the previous model, we henceforth call that model *Gaussian sparse coding*.

3. Exponential family sparse coding

Given a text document (input vector) x, we use the standard exponential family model: $P(x|b,s) = h(x)\exp(\eta^T T(x) - a(\eta))$ with the natural parameter $\eta = \sum_j b_j s_j$. Here the functions h, T and a specify the exact exponential family distribution being used (e.g., $h(x) = e^{-\|x\|^2/2}/(2\pi)^{k/2}$, $T(x) = x$, $a(\eta) = \eta^T\eta/2$ lead to a Gaussian distribution with covariance I). This includes the Gaussian sparse coding model as a special case, and we can now use a Poisson distribution if the input is integer-valued, or a Bernoulli distribution if the input is binary.

With this generative model, the basis vectors b can again be learnt from unlabeled data by solving the MAP optimization problem, or equivalently:[2]

$$\begin{aligned}\min_{B,\{s^{(i)}\}} \quad & \textstyle\sum_i \Big(-\log h(x^{(i)}) - {s^{(i)}}^T B^T T(x^{(i)}) + a(Bs^{(i)})\Big) \\ & \qquad +\beta \textstyle\sum_{i,j} |s_j^{(i)}| \\ \text{s.t.} \quad & \qquad \|b_j\|^2 \le c,\ \forall j = 1,...,n. \end{aligned} \tag{1}$$

Inspite of the generalization, this problem is still convex separately in b and s (though not jointly).[3] This again suggests an alternating minimization procedure iterating the following two steps till convergence: (i) fix the activations s, and compute the optimal basis vectors B; and, (ii) fix these basis vectors B, and compute the optimal activations s.

Step (i) involves a constrained optimization problem over B with a differentiable objective function. We can thus apply projective gradient descent updates, where at each iteration we perform a line search along the direction of the (negative) gradient, projecting onto the feasible set before evaluating the objective function during the line search. In our case, the projection operation is especially simple: we just need to rescale each basis vector to have norm c if its norm is greater than c. In our experiments, we find that such a projective gradient descent scheme is sufficiently fast for basis learning. We thus focus now on the algorithm for computing the optimal activations in Step (ii).

Step (ii) computes the optimal activation s given fixed basis vectors. The resulting problem involves a non-differentiable L_1-regularized objective function, for which several sophisticated algorithms have been developed, including specialized interior point methods (Koh et al., 2007) and quasi-Newton methods (Andrew & Gao, 2007; Yu et al., 2008). When used for computing activations with 1000 basis vectors, these methods find the optimal solution in a few seconds *per unlabeled example*. Since we often need to solve for tens of thousands of unlabeled examples repeatedly in the inner loop of the overall alternating minimization procedure, these solvers turn out to be too slow for high-dimensional problems with many basis vectors. We now present an alternative, efficient algorithm for computing the activations.

4. Computing optimal activations

We first note that since the optimal values for the activation vectors $s^{(i)}$ do not depend on each other, and can be optimized separately, it is sufficient to consider

[2] We use matrix notation $B = [b_1 b_2 \dots b_n]$.

[3] This follows from the fact that $a(\eta)$ must be convex in η (McCullagh & Nelder, 1989).

the following optimization problem for a single input x and its activation vector s:

$$\min_s \quad \ell(s) + \beta\|s\|_1 \tag{2}$$

where s corresponds to a vector of activations, and $\ell(s)$ is a specific convex function of s.

In the case of Gaussian sparse coding, $\ell(s)$ is simply a quadratic function, and the optimization problem is an L_1-regularized least squares problem that can be solved efficiently (Efron et al., 2004; Lee et al., 2007). This suggests an iterative algorithm for general exponential family distributions: at each iteration, we compute a local quadratic approximation $\hat{\ell}(s)$ to the function $\ell(s)$, and optimize the objective function $\hat{\ell}(s) + \beta\|s\|_1$ instead.[4] Using a similar insight, Lee et al. proposed the IRLS-LARS algorithm for the case of L_1-regularized logistic regression, using Efron et al.'s LARS algorithm in the inner loop to solve the approximated problem.

This method can be applied to other L_1-regularized optimization problems for which a local quadratic approximation can be efficiently computed. Indeed, for the case of the L_1-regularized exponential family in Equation (1), we can show that the local quadratic approximation at a point s is given by:

$$\hat{\ell}(s') \quad = \quad \left\|\Lambda^{1/2}Bs' - \Lambda^{1/2}z\right\|^2 \tag{3}$$

where $\Lambda_{ii} = a''\left((Bs)_i\right)$ for a diagonal matrix Λ, and $z = \Lambda^{-1}(T(x) - a'(Bs)) + Bs$.[5]

We further note that if the objective function $\ell(s)$ is reasonably approximated by a quadratic function, the solutions to the successive quadratic approximations should be close to each other. However, the LARS algorithm used in IRLS-LARS cannot be initialized at an arbitrary point, and thus has to rediscover the solution from scratch while solving each successive approximation. On the other hand, the "feature-sign search" algorithm (originally proposed in the context of Gaussian sparse coding) can be initialized at an arbitrary point (Lee et al., 2007), and can thus potentially solve the successive approximations much faster. We propose to use the feature-sign search algorithm to optimize each quadratic approximation.

The final algorithm, which we call IRLS-FS, is described below. The algorithm is guaranteed to converge to the global optimum in a finite number of iterations. (Proof similar to IRLS-LARS.)

[4]This procedure is an instance of a more general method that is sometimes called Iteratively Reweighted Least Squares (IRLS) in the literature (Green, 1984).

[5]To show that this is the local quadratic approximation to $\ell(s)$, it suffices to show that this has the same function value, gradient and Hessian at s. Indeed, we have $\nabla\ell = \nabla\hat{\ell} = -B^T T(x) + B^T a'(Bs)$, and $\nabla^2\ell = \nabla^2\hat{\ell} = B^T\Lambda B$.

Algorithm 1: IRLS-FS algorithm

Input: $B \in \mathbb{R}^{k\times n}$: matrix, $x \in \mathbb{R}^k$: vector.
Initialize $s := \vec{0}$.
while stopping criterion is not satisfied **do**
  $\Lambda_{ii} := a''\left((Bs)_i\right)$ (for diagonal matrix Λ)
  $z := \Lambda^{-1}(T(x) - a'(Bs)) + Bs$.
  Initializing feature-sign search at s, compute:
    $\hat{s} := \arg\min_{s'} \left\|\Lambda^{1/2}Bs' - \Lambda^{1/2}z\right\|^2 + \beta\|s'\|_1$
  Set $s := (1-t)s + t\hat{s}$, where t is found by a backtracking line-search that minimizes the original objective function in problem (1). (See Boyd & Vandenberghe, 2004 for details)
end while

5. Computational Efficiency

We compare the IRLS-FS algorithm against state-of-the-art algorithms for optimizing the activations, focusing on the case of binary sparse coding (i.e., $x \in \{0,1\}^k$). This case is especially interesting because this leads to an L_1-regularized logistic regression problem.[6] This problem is of general interest (e.g., see Ng, 2004), and customized algorithms have also been developed for it.

We compare the algorithm with four recent algorithms: the IRLS-LARS algorithm (Lee et al., 2006) and the l1-logreg interior point method (Koh et al., 2007) specifically for logistic regression, and the OWL-QN (Andrew & Gao, 2007) and SubLBFGS (Yu et al., 2008) algorithms for L_1-regularized convex optimization problems.[7] All algorithms were evaluated on nine L_1-regularized logistic regression problems, which arise when computing activations for text data. The sparsity parameter β was set to produce roughly 20 nonzero activations per example on average. We measured the running time taken by each algorithm to converge within a specified tolerance of the optimal

[6]We note that in each L_1-regularized optimization problem produced by exponential family sparse coding, the number of "features" is equal to the number of basis vectors used, but is *independent* of the dimensionality of inputs x in the original problem. For example, when applied to text, the number of "features" is equal to the number of basis vectors, but is independent of the number of words in the vocabulary, which could be large.

[7]Baseline algorithms: Lee et al. show that IRLS-LARS outperforms several previous algorithms, including grafting (Perkins & Theiler, 2003), SCGIS (Goodman, 2004), GenLasso (Roth, 2004) and Gl1ce (Lokhorst, 1999). IRLS-FS, IRLS-LARS and l1-logreg were implemented in Matlab, and OWL-QN and SubLBFGS were compiled in C++ with optimization flags. Code for all baselines was obtained from the respective authors.

Dataset	Small1	Small2	Small3	Medium1	Medium2	Medium3	Large1	Large2	Large3
IRLS-LARS	4.6	4.9	4.3	12.8	12.5	13.2	1131	1270	1214
l1-logreg	18.3	18.9	17.7	181	188	185	3277	3101	3013
OWL-QN	7.1	7.0	10.3	27.1	31.4	25.6	1018	739	906
SubLBFGS	33.0	22.3	23.1	101	142	57.2	1953	2717	1627
IRLS-FS	**2.5**	**2.3**	**2.2**	**5.3**	**5.5**	**5.4**	**117**	**108**	**109**

Table 1: Total running time in seconds for computing activations for 50 input examples for 9 problems (one per column). There are 3 problems each of 3 different sizes, and they are labeled Small1 to Small3, Medium1 to Medium3, or Large1 to Large3 based on the size of the problem. The Small problems had 200 basis vectors and input dimension 369, Med problems had 600 basis vectors and dimension 369, and Large problems had 1000 basis vectors and dimension 3891.

Dataset	Col	Alon	Duln	Duer	Arr	Mad	Hep	Spf	Prom	Wbc	Ion	Spl	Spc	Spam
IRLS-LARS	2.1	3.3	6.2	35.6	2.2	25.6	0.5	5.0	2.1	5.0	3.5	18.3	2.6	57.8
l1-logreg	18.3	16.8	13.6	14.4	34.8	509	1.0	3.0	2.0	3.8	2.7	12.8	2.0	**37.1**
OWL-QN	27.4	29.4	16.9	79.6	7.7	634	**0.1**	3.4	**0.4**	13.4	**1.9**	**7.1**	**0.9**	39.3
SubLBFGS	114	80.8	60.5	311	24.3	261	0.7	9.3	2.7	14.4	4.5	13.4	2.1	43.0
IRLS-FS	**1.9**	**1.9**	**2.5**	**7.1**	**1.5**	**14.0**	0.3	**2.3**	1.3	**2.9**	2.0	10.4	1.9	50.8

Table 2: Total running time in seconds for 50 trials of learning parameters of various L_1-regularized logistic regression benchmarks. The sparsity parameter β was picked to optimize generalization error of the resulting classifier. The datasets are ordered from left-to-right by increasing fraction of nonzero parameter values at the optimal solution (e.g., the problem Col had 0.2% nonzeros, Mad: 3.2%, Hep: 26.3%, Spam: 66.7%).

objective value.[8]

Table 1 shows the running times computed over 50 trials. IRLS-FS outperforms the other algorithms on this task, showing that it is well-suited to exponential family sparse coding. When a large number of basis vectors are used (while keeping the number of nonzero activations fixed), IRLS-FS can be 5 to 7 times faster than the best baseline algorithm.

This poses the question: can IRLS-FS be a useful algorithm for general L_1-regularized optimization problems (not necessarily ones generated by the sparse coding problem)? We compare the algorithms above on 14 moderate-size benchmark classification datasets, and apply L_1-regularized logistic regression to them.[9] The value of β on each benchmark was picked to optimize the generalization error of the resulting logistic regression classifier; unlike the earlier experiment, β was not set explicitly to obtain sparse solutions. Table 2 shows the running time of the algorithms to compute the optimal parameters. IRLS-FS outperforms the other algorithms on 8 out of 14 of these benchmark datasets; as expected, it performs best when the optimal parameters have few nonzero values.

[8]Details: Since IRLS-LARS solves the dual or Lasso version of our problem (i.e., with a constraint C on the L_1 norm of the activations rather than a penalty β), we follow Lee et al.'s method of using the KKT conditions to convert between the constraint value C and the equivalent penalty β for that problem. We ran all algorithms until they reached an objective value of $(1+\epsilon)f^{opt}$ where f^{opt} is the optimal objective value (we used $\epsilon = 10^{-6}$).

[9]These datasets were used to evaluate IRLS-LARS and were obtained from the authors (Lee et al., 2006).

6. Self-taught learning for text documents

The model presented in this paper generalizes Gaussian sparse coding. It is also closely related to exponential family PCA (Collins et al., 2001), which corresponds to setting the sparsity penalty β to zero, and additionally constraining the basis matrix B to have orthogonal columns. We now show that the exponential family sparse coding model can produce better self-taught learning performance than either of these previous methods.

We apply our algorithm to two self-taught learning problems in text classification: one using binary-valued input vectors $x \in \{0,1\}^k$, and another using integer-valued (word count) vectors $x \in \{0,1,2,\dots\}^k$.

We constructed five different webpage classification problems, and a newsgroup classification problem. We used 470,000 unlabeled news articles (from the Reuters corpus) to learn basis vectors according to the binary and Poisson sparse coding models.[10] Table 3 gives ex-

[10]Details: The webpage classification problems were created using subcategories of the Arts, Business, Health, Recreation and Sports categories of the DMOZ hierarchy. Each of them consisted of 10 separate binary classification problems over a 500 word vocabulary, with stopword removal and stemming. The newsgroup classification problem consisted of 10 binary classification problems constructed using the 20 newsgroups dataset. We used 600 basis vectors, and picked β to achieve roughly 10% nonzero activations. For learning, we used stochastic updates with mini-batches of 2000 randomly sampled examples, and assumed that the basis vectors had converged when the objective function did not decrease for 10 consecutive mini-batch iterations.

free	share	pharmaceut	estat	subscrib
market	exchange	drug	real	online
polici	stock	product	sale	servic
power	secur	share	loss	server
peopl	commiss	stock	loan	databas
paint	actor	novel	audio	singer
pictur	actress	literari	video	pop
portrait	film	poet	dvd	releas
museum	comedi	fiction	digit	fan
rule	star	univers	softwar	busi

Table 3: Ten examples of basis vectors trained on the Reuters data using Poisson sparse coding. Each group of five words were the most highly active words (i.e., had the highest weight) for some basis vector. Thus, each set of five words is judged to be highly correlated with each other. The top half contains basis vectors over the vocabulary for the Business category, the bottom half for the Arts category.

amples of basis vectors that were learned by applying Poisson sparse coding. Basis vectors appeared to encode related words and capture various "topics."

Using the learnt basis vectors, we computed features for each classification task using the binary and Poisson sparse coding model. We call our model "ExpSC" and compare against several baselines: the raw words themselves ("Raw"), Gaussian sparse coding ("GSC"), exponential family PCA with the same binary or Poisson exponential family assumption ("ExpPCA"), and also Latent Dirichlet Allocation (LDA), a widely-known topic model for text documents (Blei et al., 2002). All baselines (except the raw features) were trained using the same unlabeled data as our model. We also consider combinations of the raw word features with the other types of features (e.g., "Raw+ExpSC" indicates a combination of the raw features and the ExpSC features). All features were then evaluated using standard supervised-learning classifiers over 100 trials each for 4, 10 and 20 training examples.[11]

Figure 1 shows the classification results obtained for various training set sizes. The left 6 figures show results for Poisson sparse coding with varying numbers of training examples on the x-axis; the right 6 figures show results for binary sparse coding. Poisson and binary sparse coding reduce error significantly over the raw features in five and four out of six tasks respectively. The results for Poisson sparse coding are particularly striking, showing 20-30% error reduction in some cases.

Table 4 shows the average test error over all problems for the binary and Poisson case. The exponential family sparse coding features alone frequently outperform the other features, and produce slightly better results when used in combination with the raw features (ExpSC+Raw). The other three methods for using unlabeled data (GSC, ExpPCA, LDA) perform poorly in many cases.

[11] To evaluate the dependency of our results on the choice of classifier, we first evaluated many generic classifiers including SVM, Gaussian discriminant analysis (GDA), kernel dependency estimation (KDE), KNN, decision trees, etc; then, we chose the three best classifiers (GDA, KDE, and SVM) for the raw bag-of-words features. We report average results of the best performing classifier for each feature. We picked the β value used for computing the features by cross-validation. We verified that tuning the classifier hyperparameters by cross-validation does not significantly affect the results.

Discussion

In this paper, we present a general method for self-taught learning, that extends previous models in a natural way. The extensions can still be solved efficiently using the IRLS-FS algorithm, which we show to be an efficient algorithm for medium-sized L_1-regularized learning problems with sparse optimal solutions. We have shown that our model achieves better self-taught learning performance than Gaussian sparse coding or exponential family PCA. Overall, our results suggest that exponential family sparse coding can learn high-level representations of documents from unlabeled data, and that this prior knowledge is useful in document classification problems. We believe this model could be applied more generally to other language problems, including information retrieval and word sense disambiguation, where large amounts of unlabeled data are available.

Acknowledgments

We give warm thanks to Roger Grosse for useful comments. We also thank Kwangmoo Koh, Su-In Lee, Jin Yu, and Galen Andrew for providing their code. This work was supported by the DARPA transfer learning program under contract number FA8750-05-2-0249, and by ONR under award number N00014-06-1-0828.

References

Andrew, G., & Gao, J. (2007). Scalable training of L_1-regularized log-linear models. *ICML*.

Blei, D., Ng, A. Y., & Jordan, M. (2002). Latent dirichlet allocation. *NIPS*.

Boyd, S., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

Collins, M., Dasgupta, S., & Schapire, R. E. (2001). A generalization of principal component analysis to the exponential family. In *NIPS*.

Efron, B., Hastie, T., Johnstone, I., & Tibshirani, R. (2004). Least angle regression. *Ann. Stat.*, *32*, 407–499.

Goodman, J. (2004). Exponential priors for maximum entropy models. *ACL*.

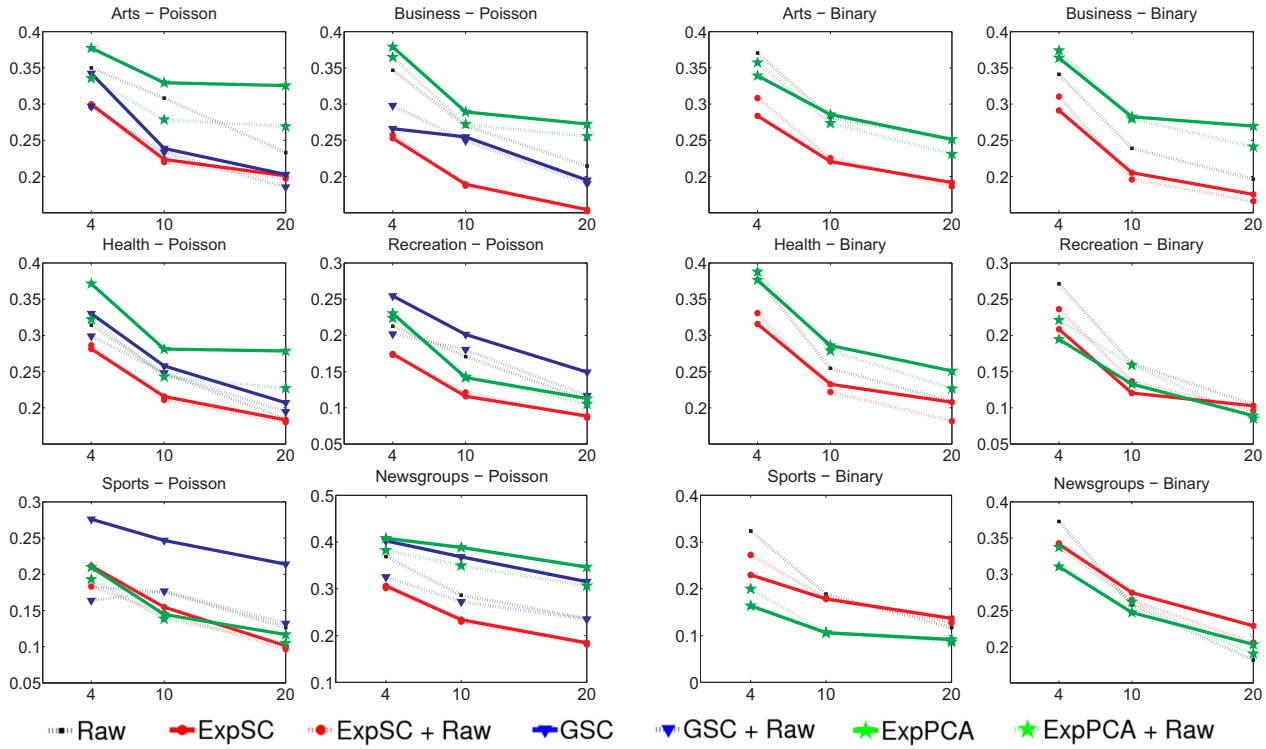

Figure 1: Test error plotted vs. training set size for the webpage and newsgroup classification tasks. Figures in the left half are for Poisson sparse coding, and in the right half for binary sparse coding. See text for explanation of algorithms, and Table 4 for averaged test errors. (Best viewed in color.)

Training set size		Raw	ExpSC	ExpSC +Raw	GSC	GSC +Raw	ExpPCA	ExpPCA +Raw	LDA +Raw
Poisson	4	29.6%	25.4%	**25.0%**	31.2%	26.4%	32.9%	30.4%	33.2%
	10	24.3%	18.9%	**18.6%**	26.1%	22.7%	26.3%	23.8%	24.7%
	20	18.4%	15.2%	**14.9%**	21.4%	17.6%	24.2%	21.2%	17.6%
Binary	4	34.3%	**27.9%**	30.0%	-	-	29.1%	31.3%	-
	10	23.0%	20.5%	**20.4%**	-	-	22.3%	22.7%	-
	20	17.7%	17.4%	**16.1%**	-	-	19.3%	17.7%	-

Table 4: Aggregate test error across all text classification problems (webpages/newsgroups), represented either word count vectors (Poisson) or using binary vectors (Binary). LDA+Raw performed better than LDA alone; so we show results only for this case.

Green, P. J. (1984). Iteratively reweighted least squares for maximum likelihood estimation, and some robust and resistant alternatives. *Journal of the Royal Statistical Society, Series B, Methodological*, *46*, 149–192.

Koh, K., Kim, S.-J., & Boyd, S. (2007). An interior-point method for large-scale L_1-regularized logistic regression. *JMLR*, *8*, 1519–1555.

Lee, H., Battle, A., Raina, R., & Ng, A. Y. (2007). Efficient sparse coding algorithms. *NIPS*.

Lee, S.-I., Lee, H., Abbeel, P., & Ng, A. Y. (2006). Efficient L_1 regularized logistic regression. *AAAI*.

Lokhorst, J. (1999). *The LASSO and Generalised Linear Models*. Honors Project, Department of Statistics, The University of Adelaide, South Australia, Australia.

McCullagh, P., & Nelder, J. A. (1989). *Generalized linear models*. Chapman & Hall.

Ng, A. Y. (2004). Feature selection, L_1 vs. L_2 regularization, and rotational invariance. *ICML*.

Olshausen, B. A., & Field, D. J. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, *381*, 607–609.

Perkins, S., & Theiler, J. (2003). Online feature selection using grafting. *ICML*.

Raina, R., Battle, A., Lee, H., Packer, B., & Ng, A. Y. (2007). Self-taught learning: Transfer learning from unlabeled data. *ICML*.

Roth, V. (2004). The generalized lasso. *IEEE Trans. Neural Networks*, *15*, 16–28.

Weston, J., Collobert, R., Sinz, F., Bottou, L., & Vapnik, V. (2006). Inference with the universum. *ICML*.

Yu, J., Vishwanathan, S. V. N., Günter, S., & Schraudolph, N. N. (2008). A quasi-Newton approach to nonsmooth convex optimization. *ICML*.