A New Algorithm for Training Sparse Autoencoders

Ali Shahin Shamsabadi*, Massoud Babaie-Zadeh*, Seyyede Zohreh Seyyedsalehi*,
Hamid R. Rabiee*, Christian Jutten†
*Sharif University of Technology, †University Grenoble Alpes
shahinshamsabadi_ali@ee.sharif.edu, mbzadeh@sharif.edu, zseyyedsalehi@ce.sharif.edu,
rabiee@sharif.edu, christian.jutten@gipsa-lab.grenoble-inp.fr

***Abstract*—Data representation plays an important role in performance of machine learning algorithms. Since data usually lacks the desired quality, many efforts have been made to provide a more desirable representation of data. Among many different approaches, sparse data representation has gained popularity in recent years. In this paper, we propose a new sparse autoencoder by imposing the power two of smoothed L0 norm of data representation on the hidden layer of regular autoencoder. The square of smoothed L0 norm increases the tendency that each data representation is "individually" sparse. Moreover, by using the proposed sparse autoencoder, once the model parameters are learned, the sparse representation of any new data is obtained simply by a matrix-vector multiplication without performing any optimization. When applied to the MNIST, CIFAR-10, and OPTDIGITS datasets, the results show that the proposed model guarantees a sparse representation for each input data which leads to better classification results.**

I. Introduction

The performance of many machine learning algorithms (e.g. classification and clustering) is heavily influenced by the quality of their input data. They show their best performance and highest efficiency only when they are provided with suitable inputs [1]. In this purpose, many efforts have been made to achieve the desirable representation of data in different machine learning applications.

In general, unknown complex statistical relationships may exist between elements of raw data [2]. Therefore, many different representation approaches can be utilized to process the raw data. Good representations are those which extract the causal factors of statistical variations that generate the data [1]. Currently, there are two main approaches to obtain representations from raw data, namely feature engineering and representation learning [1], [3], [4].

In feature engineering, which is a traditional approach, transformations are designed based on the domain of input data [5], to map data from the input space into the feature space for obtaining a new representation [1]. For example, algorithms such as SIFT [6], RIFT [7], HOG [8], GHOG [9], and SURF [10] have been proposed to extract features from images. Although these methods have carefully been designed, they are usually very time-consuming, labor-intensive, and dependent on human ingenuity and prior knowledge to decide which features should be extracted for a suitable data representation [1], [3].

This work has been partly supported by the European project ERC-2012-AdG-320684-CHESS.

In representation learning [1], researchers are looking for methods to automatically learn a representation from data to be used in higher-order tasks (e.g. classification). These methods try to eliminate the complicated algorithms used for feature extraction. As a result, these algorithms are extensible to different domains of input data [5], [11].

More recently, a large number of researchers have focused on development of new sparse techniques for data representation [12], [13], [14], [15], [16]. Sparse representation encodes a large number of data by using a small number of components, which leads to easier interpretation of the representation, robustness to variations in input, and easier classification [12].

In many applications, available data is unlabeled, and hence unsupervised algorithms are preferred [17]. The main objective of unsupervised learning methods is to learn a representation from unlabeled data by detecting and removing redundancies in the data, and keeping only the features essential for the higher-order tasks [18]. There are two unsupervised learning approaches to obtain a sparse representation; sparse coding and sparse autoencoder.

The goal of sparse coding [15] is to learn a dictionary for the input dataset such that it is possible to reconstruct each input data from a sparse weighted linear combination of basis vectors (*i.e*, the columns of the dictionary matrix) [19]. The coefficients of this linear combination provide the data representation. As stated in [20], [21], sparse coding does not have any explicit encoder, and hence optimization algorithms should be run two times on the objective function (once for calculating the representation assuming a constant dictionary, and once for learning the dictionary given a fixed representation). Moreover, a new run of the optimization algorithm is required for each new item in the test data. Therefore, this method is not suitable for real-time applications and has limited scalability.

In [22], [3] by imposing a sparsity penalty to learned features of autoencoders [23] that are unsupervised encoder-decoder neural networks, the authors have tried to tackle the aforementioned problem of sparse coding methods. The encoder part transforms data from the input space to feature space which results in sparse features, and the decoder part transforms the extracted sparse features from feature space into the input space. When the parameters of the encoder are learned, it will encode new data without utilizing an optimization algorithm, and the decoder reconstructs the input

from the representation obtained from the encoder [12]. The main issue with the sparse autoencoders is that there is no guarantee to obtain sparse representations for new data or even the training data, because in the training phase, the autoencoder is not trained to produce a sparse representation for each input data.

In this paper, a sparse autoencoder is proposed to overcome the limitations of existing sparse autoencoders, by introducing a new distributed sparse representation algorithm. In general, although the sparsity of a vector is usually defined by its L0 norm, the L0 norm is not a convenient measure of sparsity [24] because it is a discontinuous and non-smooth function. Hence, a smooth approximation of the L0 norm by using smooth Gaussian kernels [24] have been adopted in our work. To obtain a representation with relatively equal sparsity for each data vector, the square of the smooth approximation of the L0 norm [25] has been used to solve the problem of sparse autoencoders. It also guarantees the existence of a sparse representation for the training and test datasets. By using the proposed sparse autoencoder, once the network parameters are learned, the sparse representation of any new data vector is obtained simply by a matrix-vector multiplication without utilizing any additional optimization algorithm that is necessary in typical sparse coding algorithms.

In summary, the proposed method has the following features and advantages:

- An approximation of the L0 norm has been used in autoencoder which is smooth and differentiable and therefore is not too sensitive to noise.
- Obtaining the sparse representation of a vector using smoothed L0 norm is highly faster than L1 norm [24].
- The square of the smooth approximation of the L0 norm has been used to create the tendency that the representation of each data vector is "individually" sparse.
- Contrary to sparse coding algorithms that need to repeat the optimization algorithm for obtaining a sparse representation of each new data in the test phase, after training, our method computes the sparse representation of new data by a simple matrix-vector multiplication.

The rest of this paper is organized as follows. In section II autoencoder and prior works on sparse autoencoder will be explained. Section III and IV are devoted to presenting the proposed sparse autoencoder and results of experiments, respectively. Finally, concluding remarks are provided in section V.

II. Autoencoder Related Works

In this section, after a brief introduction to autoencoders, various models of sparse autoencoders are discussed.

A. Autoencoder

An autoencoder [23] (Fig. 1) is an unsupervised artificial neural network with three layers, namely, the input (visible), intermediate (hidden), and output layers. Autoencoders can reconstruct the input data in the output layer by using the representation obtained in the hidden layer. In its general form,

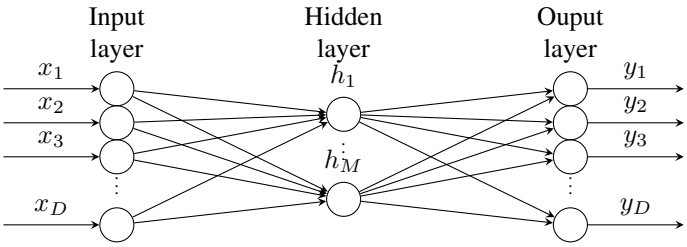


Fig. 1. A typical Autoencoder.

an autoencoder consists of two components called encoder and decoder. In the encoder part, each input $\mathbf{x} \in \mathbb{R}^D$ is mapped onto a hidden-layer representation $\mathbf{h} \in \mathbb{R}^M$ by applying a typical nonlinear function to a linear combination of elements of the input vector. Similarly, in the decoder part, the network tries to reconstruct the input, by applying a typical nonlinear function to a linear combination of the elements of the obtained representation. Thus, the parameters of autoencoder will be learned by minimizing the difference between the input and output of the network.

B. Sparse Autoencoder

The idea of sparse autoencoders is to impose a constraint on the network such that the representation has the required sparsity characteristics while reconstructing the input at the output layer. In [22], [3], an autoencoder is trained such that the distribution function of each neuron is close to the Bernoulli distribution with a small mean, in order to learn sparse data representation.

In addition to the aforementioned methods, in [3] another sparse autoencoder has been proposed that exploits the L1 norm for data representations.

III. Proposed Method

In this section, we propose a new sparsity-generating term in the cost function for training autoencoders. For this purpose, in III-A, the rationale for using the L0 norm will be explained and then the idea of using its continuous approximation will be proposed. In III-B, the idea of using the square of the L0 norm will be proposed. In III-C, the sparsity-generating term is introduced and the equations required for learning the autoencoder parameters are calculated.

A. Continuous approximation of L0 norm

Among infinitely many representations of the data, we are looking for a representation that has the characteristic of being sparse, which results in the following optimization problem:

$$\min \|\mathbf{h}\|_P^P \qquad \text{s.t} \qquad \mathbf{x} = \mathbf{y}, \tag{1}$$

where:

$$\mathbf{y} = g(\mathbf{W}^{(2)}\mathbf{h} + \mathbf{b}^{(2)}), \qquad \mathbf{h} = f(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}), \tag{2}$$

variables $\mathbf{x}$ and $\mathbf{y}$ stand for a single input and output of the proposed autoencoder and $\mathbf{h}$ is the representation of the data. $\mathbf{W}^{(1)} \in \mathbb{R}^{M \times D}$ is the coefficient matrix between the input and

the intermediate layer, $\mathbf{b}^{(1)} \in \mathbb{R}^M$ is the input layer bias, f is the activation function of the hidden layer neurons, $\mathbf{W}^{(2)} \in \mathbb{R}^{D\times M}$ is the coefficient matrix between the hidden and the output layer, $\mathbf{b}^{(2)} \in \mathbb{R}^D$ is the hidden layer bias, and g is the activation function of the neurons in the output layer.

Next, we look into the characteristics of various norms and select a norm that leads to a better sparse representations. In general, the smaller values of P results in greater probability that the contours of $\|\mathbf{h}\|_P^P$, and the plane $\mathbf{x} = \mathbf{y}$ coincide on the coordinate axes. Therefore, the solution with greatest sparsity corresponds to the L0 norm.

The L0 norm indicates the number of the non-zero elements of $\mathbf{h}$, therefore the L0 norm of a vector is a discontinuous function, and minimizing the L0 norm is an NP-hard problem [26]. In [24], the L0 norm has been approximated by using a smoothed L0 norm. In the proposed method, we have utilized the same method at the output of neurons in the hidden layer and the L0 norm of $\mathbf{h}$ (outputs of the encoder neurons) is:

$$\|\mathbf{h}\|_0 \approx M - \lim_{\sigma\to 0}\sum_{i=1}^{M}\exp(\frac{-h_i^2}{2\sigma^2}). \tag{3}$$

B. Sparseness of all data

For learning the parameters of the autoencoder, the network is trained with a training dataset for solving the optimizaion problem:

$$\min \sum_j \|\mathbf{h}^{(j)}\|_0 \qquad \text{s.t} \qquad \mathbf{X} = \mathbf{Y}, \tag{4}$$

where the matrix $\mathbf{X} = [\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(N)}]$ collects all the data, and:

$$\mathbf{H} = f(\mathbf{W}^{(1)}\mathbf{X} + \mathbf{B}^{(1)}), \quad \mathbf{Y} = g(\mathbf{W}^{(2)}\mathbf{H} + \mathbf{B}^{(2)}). \tag{5}$$

However, the above sparse autoencoder does not sparsely represent all the data. As an example, assume that the training set includes 5 data points, and we are trying to obtain a sparse representation of size 4 by optimizing the cost function of the sparse autoencoder in (4). The matrix of representation $\mathbf{H}$, that is obtained for the data between $\mathbf{H}_1$ and $\mathbf{H}_2$ by optimizing (4) is $\mathbf{H}_1$. This result shows that for the data items 3, 4, and 5, the representation is over sparse, while for the data items 1 and 2 the representation is not sparse.
This is not the desired answer, which should provide an equal degree of sparsity for each of the data items.

$$\mathbf{H}_1 = \left(\begin{bmatrix}0&1&1&0&0\\1&1&0&0&0\\1&0&0&1&0\\1&1&0&0&1\end{bmatrix}\right) \mathbf{H}_2 = \left(\begin{bmatrix}1&0&1&0&0\\0&1&1&0&1\\1&1&0&1&0\\0&0&0&1&1\end{bmatrix}\right)$$

To alleviate this problem, as proposed in [25], we increase the power of L0 norm from 1 to 2. This will cause the non-zero elements to be penalized more, and therefore creating a tendency to distribute the sparsity among all data points. In the above example, the cost function:

$$\min \sum_j \|\mathbf{h}^{(j)}\|_0^2 \qquad \text{s.t} \qquad \mathbf{X} = g(\mathbf{W}^{(2)}\mathbf{H} + \mathbf{B}^{(2)}), \tag{6}$$

will prefer $\mathbf{H}_2$ over $\mathbf{H}_1$.

C. Proposed sparse autoencoder

The proposed sparsity term of data representation in autoencoders, which uses the square of the continuous approximation of the L0 norm, is given by cost C_s (s for sparse):

$$C_s(\mathbf{h}) = \sum_{j=1}^{N}\Big(M - \lim_{\sigma\to 0}\sum_{i=1}^{M}\exp\big(\frac{-(h_i^{(j)})^2}{2\sigma^2}\big)\Big)^2, \tag{7}$$

where $h_i^{(j)}$ is output of the i-th neuron of hidden layer in response to the j-th input data item and is defined as:

$$h_i^{(j)} = f(\mathbf{W}_{(i)}^{(1)}\mathbf{x}^{(j)} + \mathbf{b}_{(i)}^{(1)}). \tag{8}$$

Finally, the proposed cost function of the sparse autoencoder, C_t (t for total), is obtained by adding the constraint of equation (6) as a penalty term, and is given by:

$$C_t(\mathbf{h}) = \frac{1}{2N}C_s(\mathbf{h}) + \frac{\lambda}{2N}C_r(\mathbf{x},\mathbf{y}), \tag{9}$$

where the reconstruction cost C_r is:

$$C_r(\mathbf{x},\mathbf{y}) = \sum_{j=1}^{N}||\mathbf{x}^{(j)} - \mathbf{y}^{(j)}||_2^2, \tag{10}$$

where N denotes the total size of the training data set, and λ is the hyper-parameter that controls the trade-off between the reconstruction error and sparsity.

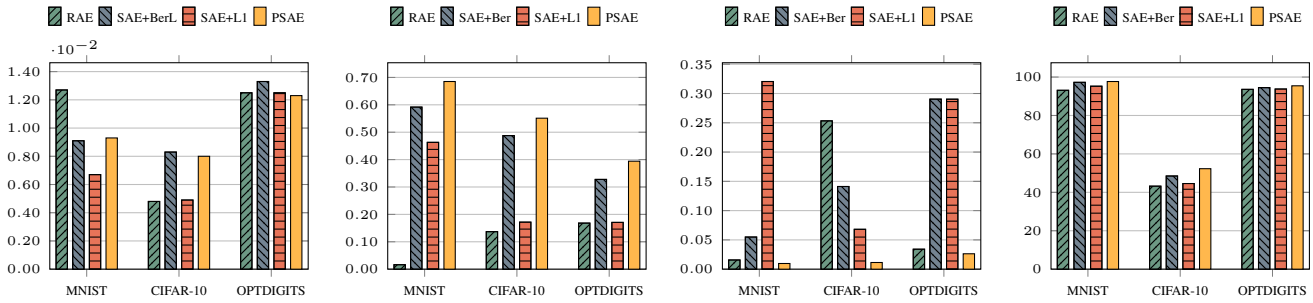
For learning the network parameters, it is necessary to minimize $C_t(\mathbf{h})$ with respect to the parameters. To utilize gradient based optimization algorithms, the gradient of the $C_t(\mathbf{h})$ with respect to the network parameters should be calculated. For updating the parameters of the decoder part $(\mathbf{W}^{(2)}, \mathbf{b}^{(2)})$ only the $C_r(\mathbf{x},\mathbf{y})$ plays a role in the modification of the parameters of the decoder part, and $C_s(\mathbf{h})$ has no effect on updating these parameters, because the sparsity-generating term is defined for the hidden layer and is independent of the parameters pertaining to the interaction between the hidden layer and the output layer. However, it is clear that both terms of $C_r(\mathbf{x},\mathbf{y})$, and $C_s(\mathbf{h})$ affect the updating of the encoder parameters.

The gradient of the $C_s(\mathbf{h})$ with respect to $\mathbf{W}^{(1)}$ is a matrix in which the (i,j)-th element is the derivative of $C_s(\mathbf{h})$ with respect to the (i,j)-th element of the matrix $\mathbf{W}^{(1)}$, and the i-th row of the gradient of $C_s(\mathbf{h})$ can be calculated from:

$$\sum_{j=1}^{N}\Bigg(\Big(M - \sum_{l=1}^{M}\exp\big(\frac{-(h_l^{(j)})^2}{2\sigma^2}\big)\Big)\frac{(h_i^{(j)})^2}{2\sigma^2} \exp\big(\frac{-(h_i^{(j)})^2}{2\sigma^2}\big)\acute{h}_i^{(j)}(\mathbf{x}^{(j)})^T\Bigg). \tag{11}$$

In addition, $\nabla_{\mathbf{b}^{(1)}}C_s(\mathbf{h})$ is a vector whose i-th element is the derivative of $C_s(\mathbf{h})$ with respect to the i-th element of the vector $\mathbf{b}^{(1)}$ and is calculated from:

$$\sum_{j=1}^{N}\Bigg(\Big(M - \sum_{l=1}^{M}\exp\big(\frac{-(h_l^{(j)})^2}{2\sigma^2}\big)\Big)\frac{(h_i^{(j)})^2\acute{h}_i^{(j)}}{2\sigma^2}\exp\big(\frac{-(h_i^{(j)})^2}{2\sigma^2}\big)\Bigg). \tag{12}$$



(a) Reconstruction Error (b) Sparseness (c) Sparsity Standard Deviation (d) Classification Accuracy

Fig. 2. Compare the Regular Autoencoder (RAE), Sparse Autoencoder with L1 penalty on the data representation (SAE+L1) [3], Sparse Autoenocer with Bernoulli distribution for data representation (SAE+Ber) [3], and Proposed Sparse Autoencoder (PSAE)

IV. Experiments

In this section, first the datasets and the criteria for assessing the performance of the proposed method are described. Then, the structural characteristics of the implemented models are explained. Finally, the results are presented and analyzed.

A. Datasets and Evaluation Criteria

To measure the performance of the proposed algorithm, the MNIST [27], CIFAR-10 [28] and OPTDIGITS image datasets are used.

To evaluate the learned representations, we consider the following four criteria.

1) Reconstruction Error: Reconstruction error (RE) for data $\mathbf{x}^{(\mathrm{j})}$ reconstructed as $\mathbf{y}^{(\mathrm{j})}$, is given by [29]:

$$\mathrm{RE} = \frac{1}{D}\sum_{i=1}^{D} \|x_i^{(j)} - y_i^{(j)}\|^2 \tag{13}$$

where D is the dimension of the input and output data.

2) Sparseness: Sparseness (S) is defined as [30]:

$$\mathrm{S}(\mathbf{h}) = \frac{\sqrt{M} - \frac{\sum_{i=1}^{M}|h_i|}{\sqrt{(\sum_{i=1}^{M}|h_i|^2)}}}{\sqrt{M}-1}, \tag{14}$$

where for vector $\mathbf{h}$, a real number between zero and one is obtained as its sparseness. Closer values to one correspond to more sparsness.

3) Sparsity Standard Deviation: We introduce a new criterion, called Sparsity Standard Deviation (SSD), to evaluate the performance of an algorithm in sparse representation for each individual vector. The SSD is defined as:

$$\mathrm{SSD}(\mathbf{H}) = \sqrt{\frac{1}{N}\sum_{j=1}^{N}\left(\mathrm{S}(\mathbf{h}^{(j)}) - \left(\frac{1}{N}\sum_{j=1}^{N}\mathrm{S}(\mathbf{h}^{(j)})\right)\right)}. \tag{15}$$

Lower values of this criterion indicates that the sparseness of the individual data items are closer to each other.

4) Classsification Accuracy: More accurate classification based on the representation of the data is an indicator of more distinguishing features from the input data in the representation.

B. Structure of The Proposed Model

For learning the representations of data with autoencoder, we utilized a structure that consists of three stages as explained in the following.

Before feeding the data into the representation learning model, it should go through a normalization process. The normalization results in better convergence rates of the representation learning algorithm [3]. In the normalization process, first the mean and standard deviation of each dimension of the data is normalized to zero and one, respectively. Then, the pixel values are mapped into the (0,1) interval. Then, to obtain the data representation, the parameters of the model are learned by using the normalized training dataset as both input and desired output of the autoencoder. More precisely:

- The normalized training dataset is given to the autoencoder as both the input and the desired output, and the proposed autoencoder cost function is optimized according to (9) to learn the parameters of the model.
- To escape local minima as in [24], we use a Graduate Non-Convexity (GNC) approach, that is, a proper decreasing sequence is selected as the parameter σ for the sparsity term proposed in (7) in the form $\{\sigma_1, \sigma_2, ..., \sigma_K\}$.

For σ_1, the model parameters are initialized randomly. For other terms of the sequence, the model is trained using the optimization algorithm in a few iterations. The initial value of the parameters of the model, σ_i, is obtained from the final value of the parameters according to σ_{i-1}. The representations of the test data items are obtained by using the encoder parameters learned at the learning stage.

Finally, the representation learned by using the autoencoder is fed into the classifier as input. The classifier (a one-layer neural network with the softmax activation function) is trained using the representations of the data. The implementation strategies are the same as strategies in [3]. The results of the models based on these criteria are shown in Fig. 2.

C. Analysis of Experiments

The small values obtained for RE in all four methods, which is shown in Fig. 2a, demonstrates the capability of autoencoders in reducing RE. As one might expect, the RE

of the regular autoencoder is smaller than the other methods in most experiments. The reason for this is that in a regular autoencoder, no constraint is imposed on the representation of the data and as a result, the data items can be reconstructed from their representation with less error. After the regular autoencoder, the least RE belongs to the proposed autoencoder. This shows that the representation preserves significant information from the data, which results in the reconstruction of the data with the least amount of error.

Another point about the results of this experiment is the sparseness of the learned representation which is shown in Fig. 2b. As expected, in the proposed method learning a sparse representation of the data is made possible without much loss of classification accuracy or increase in RE. Considering the results pertaining to the SSD (Fig. 2c) of the learned representations reveals that the proposed method results in representations with almost equal S for all of the data items. The results presented in the Fig. 2d indicate that the sparse representation of the data has resulted in an improved efficiency in classification. Hence, a sparse autoencoder result in higher CA than regular autoencoders. In addition, the greatest CA using the representations of the data belongs to the proposed algorithm. This means that the proposed algorithm for representation learning has the capability of extracting more distinguishing features from the input data as a result of distributing the number of zero elements evenly across all of the data representations.

V. Conclusion

In this paper, we proposed a new sparse autoencoder which promotes sparse representation for each data, individually. In addition, a new criteria for evaluating the ability of models in learning sparse representation for each data was presented. While the proposed method results in sparser representations for all the data, with almost equal numbers of non-zero elements, it also improves the classification accuracy.

References

[1] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 8, pp. 1798–1828, 2013.

[2] Y. Bengio, "Learning deep architectures for ai," *Foundations and trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[3] N. Jiang, W. Rong, B. Peng, Y. Nie, and Z. Xiong, "An empirical analysis of different sparse penalties for autoencoder in unsupervised feature learning," in *Neural Networks (IJCNN), 2015 International Joint Conference on*. IEEE, 2015, pp. 1–8.

[4] A. Ng, "Sparse autoencoder," *CS294A Lecture notes*, vol. 72, pp. 1–19, 2011.

[5] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng, "Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011, pp. 3361–3368.

[6] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[7] S. Lazebnik, C. Schmid, and J. Ponce, "Semi-local affine parts for object recognition," in *British Machine Vision Conference (BMVC'04)*. The British Machine Vision Association (BMVA), 2004, pp. 779–788.

[8] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 886–893.

[9] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 10, pp. 1615–1630, 2005.

[10] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Computer vision–ECCV 2006*. Springer, 2006, pp. 404–417.

[11] S. J. Pan and Q. Yang, "A survey on transfer learning," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 22, no. 10, pp. 1345–1359, 2010.

[12] Y.-l. Boureau, Y. L. Cun *et al.*, "Sparse feature learning for deep belief networks," in *Advances in neural information processing systems*, 2008, pp. 1185–1192.

[13] C. Poultney, S. Chopra, Y. L. Cun *et al.*, "Efficient learning of sparse representations with an energy-based model," in *Advances in neural information processing systems*, 2006, pp. 1137–1144.

[14] E. Doi, D. C. Balcan, and M. S. Lewicki, "A theoretical analysis of robust coding over noisy overcomplete channels," in *Advances in neural information processing systems*, 2005, pp. 307–314.

[15] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by v1?" *Vision research*, vol. 37, no. 23, pp. 3311–3325, 1997.

[16] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *Image Processing, IEEE Transactions on*, vol. 15, no. 12, pp. 3736–3745, 2006.

[17] Y. B. Ian Goodfellow and A. Courville, "Deep learning," 2016, book in preparation for MIT Press. [Online]. Available: http://www.deeplearningbook.org

[18] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction," in *Artificial Neural Networks and Machine Learning–ICANN 2011*. Springer, 2011, pp. 52–59.

[19] H. Lee, A. Battle, R. Raina, and A. Y. Ng, "Efficient sparse coding algorithms," in *Advances in neural information processing systems*, 2006, pp. 801–808.

[20] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 1794–1801.

[21] K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "Fast inference in sparse coding algorithms with applications to object recognition," *arXiv preprint arXiv:1010.3467*, 2010.

[22] H. Lee, C. Ekanadham, and A. Y. Ng, "Sparse deep belief net model for visual area v2," in *Advances in neural information processing systems*, 2008, pp. 873–880.

[23] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[24] H. Mohimani, M. Babaie-Zadeh, and C. Jutten, "A fast approach for overcomplete sparse decomposition based on smoothed norm," *IEEE Transactions on Signal Processing*, vol. 57, no. 1, pp. 289–301, 2009.

[25] Z. Sadeghipoor, M. Babaie-Zadeh, and C. Jutten, "Dictionary learning for sparse decomposition: a new criterion and algorithm," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 5855–5859.

[26] B. K. Natarajan, "Sparse approximate solutions to linear systems," *SIAM journal on computing*, vol. 24, no. 2, pp. 227–234, 1995.

[27] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[28] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," 2009.

[29] A. Lemme, R. F. Reinhart, and J. J. Steil, "Efficient online learning of a non-negative sparse autoencoder." in *ESANN*. Citeseer, 2010.

[30] P. O. Hoyer, "Non-negative matrix factorization with sparseness constraints," *Journal of machine learning research*, vol. 5, no. Nov, pp. 1457–1469, 2004.