Knowledge discovery in databases based on deep neural networks

Yuanhua Tan[1], Chaolin Zhang[1], Yonglin Ma[2], and Yici Mao[3]
[1]Karamay Hongyou Software Co., Xinjiang, 834000, China
[2]Application Management Office of SINOPEC IT Management Department, Beijing, 100728, China
[3]Karamay Municipal People's Government Bureau of Information Industry, Xinjiang, 834000, China
Email: tanyh66@petrochina.com.cn (Y. Tan), lin_2728@126.com (C. Zhang),
mayl@sinopec.com (Y. Ma), 18909901617@126.com (Y. Mao)

***Abstract*—Knowledge discovery in databases (KDD) has received great progress in recent years for the need of mining useful knowledge in the ever growing information. The advances in machine learning technologies effectively promote KDD in the procedures of feature extraction and data categorization. This paper introduces a framework that combines feature extraction and categorization of the collected data in order to recognize useful structured patterns that underlies the raw data. This frame work consists of three modules: data pre-processing module, feature extraction module, and feature classification module. We propose a four-layered deep neural network as the feature extraction architecture. Each layer is trained in an unsupervised way as one auto-encoder with sparsity constraint. We employ a softmax classifier to assign a label to the extracted feature. The supervised and unsupervised training strategies are discussed at the end of this paper to disambiguate the training procedure of the entire model.**

***Index Terms*—Knowledge discovery, deep neural network, sparse auto-encoder, softmax classification**

I. Introduction

The last decades witnessed a great progress in information technology. The amount of information that is delivered, transmitted or processed increases at a fast pace. People are buried in the ocean of information. Knowledge discovery in databases(KDD) is a technique that has attracted considerable researchers to discover knowledge that is useful to people from information [1]. The knowledge is referred to as the information that represents the inner connection of subjects, underlies the raw data, and is easy to be accepted [2]. To deal with the ever increasing market competition and need for decision support, many manager of the enterprises resort to knowledge discovery in databases as a powerful tool for decision making, demand analysis, etc [3], [4].

Previous methods are well-suited for problems that have limited amount of data, or fixed size data. However, they are less efficient for larger databases. Hence, research work has been dedicated to finding an appropriate data arrangement, an discriminative representation, and a expressive model [5]. Researchers have found that the above mentioned three factors are highly correlated. A key factor in KDD is data mining, which incorporates database system, statistics, machine learning, neural networks, logistic theory, rough sets, etc. A proposed discovery strategy can increase the efficient of data mining, as well as reduce human intervention [6].

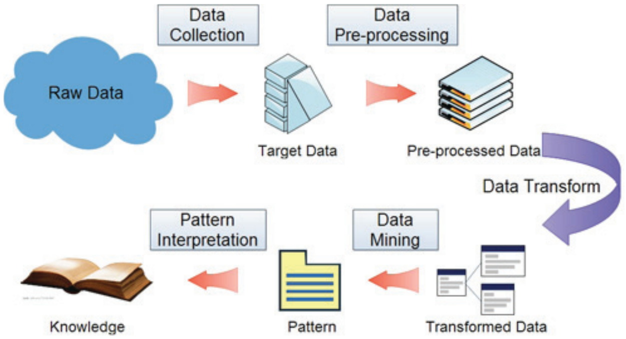

Fig. 1. Common knowledge discovery in databases procedure.

A common knowledge discovery procedure is illustrated in Figure 1, in which we can see that it consists of 5 parts, i.e. data collection, data pre-processing, data transform, data mining, and pattern interpretation. Data collection means creating a database from immense information in real world according to the user's specific interest. The raw data in the world often exist in a chaotic order. The data collection is accomplished by human labor and the data usually comprise noise, which need the data pre-processing procedure to remove the noise. Data pre-processing employs techniques such as low-pass/high-pass filtering for signal data, median/average filtering for image data, and Gaussian filtering for audio data. Among the collected data, there are also many instances that come from other domain, and that have a long distance to the desired data according to a specific distance metric. These instances are usually regarded as the outliers, which also need to be removed in the data pre-processing procedure. Since the model for data mining varies in different circumstances, the input data vary in format accordingly, which needs the data transform to accomplish this. For example, the conventional convolutional neural networks require the input image to have a fixed size [7], while the collected images have arbitrary sizes. The kernel support vector machine performs well when the input data to be in a relatively low dimension, while the data

978-1-4799-8389-6/15/$31.00 ©2015 IEEE

collected often resides in a relatively high feature space [8].

The core of a knowledge discovery strategy is the data mining. It has a direct impact on the performance and effectiveness of the whole procedure. Most data mining methods are based on machine learning, pattern recognition, and statistics. Particularly, they can be generally categorized into the following groups:

1) Categorization. Mine for interpretation or model of each category among the statistics in the database. This is accomplished by technique such as decision tree, rough set, neural network, etc.

2) Regression analyses. Find a function that maps the data to a real number, predictive variable in order to discover the dependencies between variables and attributes.

3) Clustering. Recognize a set of clustering rules, mainly by means of feasibility density estimation, so that the data can be clustered into different groups.

4) Parsimony. Given a dataset, find a tight description of it. Most descriptions employ linked data rules, visualization techniques, and business diagrams.

One of the most widely accepted technique in data mining is machine learning [9], which investigates how to simulate or realize the learning behavior of human beings to obtain new knowledge or skills, or to organize learned knowledge structure in a new way so that the performance could be continuously updated. As the core of artificial intelligent, machine learning is a fundamental way to computer intelligence. Its applications include many aspects in artificial intelligence, employing induction and synthesis rather than deduction. Deep learning, a branch of machine learning, obtains a more powerful feature representation ability in recent years. The coming of big data era stimulates the progress of data-driven deep learning model, enabling a more effective feature extraction mechanism [10].

Deep neural network is a mathematical model of information processing that is similar to the structure of brain synaptic. Deep neural network is a kind of computational model, composed of a large set of nodes(or neurons) connected between each other. Each node represents a specific output function called the activation function. Each connection between two nodes represents a weight, which is equivalent to the artificial neural network memory. The network's output depends on the connection of the nodes, the weight and activation function. The network itself may be an approximation of some algorithms in nature, or the expression of a logical strategy.

II. Feature Extraction Strategy

In this section, we introduce the feature extraction strategy, which plays an important role in data mining. One of the most widely used strategy is to extract features from raw data in a bottom-up processing order, including low-level extraction, encoding, pooling, etc. Another effective feature extraction method is the deep neural network model that extracts features automatically.

A. Low-level feature extraction

Low-level feature extraction is the first step toward pattern recognition, including two extraction strategy: interested point extraction and dense feature extraction. Common local feature points include Scale-Invariant Feature Transform (SIFT) [11], Histogram of Oriented Gradient (HOG) [12], Local Binary Pattern (LBP) [13], etc. Methods that achieve better performance often employ a combination of feature point types in a densely extraction way. By extracting a large number of redundant features, these methods aim at capturing the low-level information in a greedy way, preventing the loss of too much useful information.

B. Feature encoding

Since the densely extracted low-level features contain a much redundancy and noise, an encoding mechanism is needed to enhance the robustness of feature expression as well as to obtain the more discriminative features. Some most famous feature encoding algorithms include vector quantization [14], kernel codebooks [15], sparse coding [16], locality-constrained linear coding [17], salient coding [18], Fisher vector coding [19], super-vector coding [20], Bag of Word model [21](Figure 2), etc.

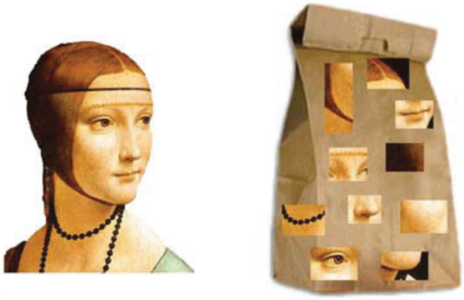

Fig. 2. Bag of Words representation of image features. Left: object presented in an image. Right: object parts in a bag.

C. Feature pooling

Feature pooling integrates the encoded feature set with spatial information. Among the encoded features, take the mean or average value of each dimension to a pooled feature. In this step we can get a more robust feature representation which is invariant to certain transforms, as well as avoid high computational cost when using the entire feature set. Spatial Pyramid Matching (SPM) [22] is proposed to averagely divide an image into several subregions, i.e. 1*1, 2*2, 4*4. Take feature pooling operation in each subregion and then concatenate the subregion feature as a feature representation. It is the dual form of Pyramid Matching Kernel (PMK) [23] in spatial region. Yang et al. [24] proposed linear spatial pyramid matching using sparse coding (ScSPM). By learning overcomplete sparse features, ScSPM maps linear non-separable features in low dimension into linear separable features in high dimension, so that linear Support Vector machine (SVM)

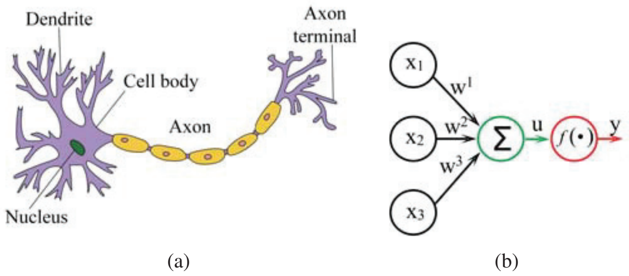

Fig. 3. Basic architectures of human neuron and artificial neuron.

can be applied on them, which greatly reducing the classifier training time and memory space consumption.

D. Auto-encoder

The auto-encoder [25] is a basic neural network structure which is proposed in twentieth century, and it is widely used for dimension reduction and feature extraction. The auto-encoder consists of two basic components: encoder and decoder. The encoder transforms the input data to the hidden layer while the decoder transforms the hidden layer to the output layer, requiring that the output data resemble the input data. Auto-encoder is an unsupervised learning method based on feature reconstruction. Added with different constraints, we can get different variations such as Denoising Auto-encoders [26] and Sparse Auto-encoders [27].

III. Multi-layer Neural Networks

A. Model architecture

Studies in neurophysiology and neuroanatomy have shown that neuron is the basic unit of brain tissue. It is structured as four main parts: cell body, dendrite, axon, and axon terminal (as shown in Figure 3(a)). The cell body is the main part of a neuron. The dendrite has exquisite tubes that extent out from the cell body, which receive information from surrounding neurons, known as the input of a neuron. The axon is the longest channel that transforms information to other neurons via its terminal, which is known as the output. Similarly, researchers have proposed a basic artificial neuron model that simulate the basic structure and function of human neuron (as shown in Figure 3(b)). It typically consists of three parts: input, summation unit, and activation function. The input data (denoted as $X = \{x_1, x_2, ..., x_n\}$), together with a bias term b, are taken as the input to the summation unit through the weights $W = \{w_1, w_2, ..., w_n\}$:

$$u = WX + b = \sum_{i=1}^{n} w_i x_i + b \tag{1}$$

Alternatively, the above summation function is usually denoted as a more uniform form:

$$u = WX = \sum_{i=0}^{n} w_i x_i \tag{2}$$

where the bias term b is denoted as w_0 and X is add by one dimension $X = \{1, x_1, x_2, ..., x_n\}$. The activation function transforms u to the final output. Many types of functions can be chosen to be the activation function. We choose the most common one, known as the sigmoid function:

$$y = sigmoid(u) = \frac{1}{1 + exp(-u)} \tag{3}$$

The proposed multi-layer neural network is structured by hooking together many of the single artificial neuron, as illustrated in Figure 4. In this figure, the circles denote the summation unit and the activation function. Input and output data are connected by a straight line. The network has four layers: input layer (X), hidden layer 1 (a^1), hidden layer 2 (a^2), and output layer (y). The information is passed through the network by the weights in each layer (W^1, W^2, W^3). Specifically, the computation that this neural network represents is given by the following equations:

$$\begin{aligned} u^1 &= W^1 X \\ a^1 &= sigmoid(u^1) \\ u^2 &= W^2 a^1 \\ a^2 &= sigmoid(u^2) \\ u^3 &= W^3 a^2 \\ y &= sigmoid(u^3) \end{aligned} \tag{4}$$

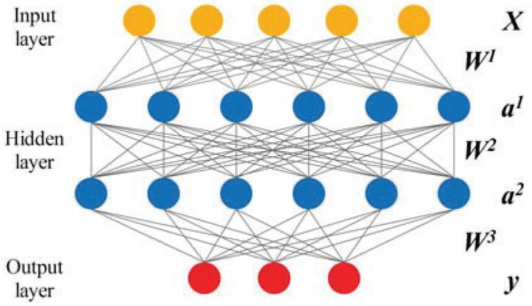

Fig. 4. Architecture of the proposed multi-layer neural network.

B. Back propagation

For a training set with labels $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), ..., (x^{(m)}, y^{(m)})$, We can obtain the weights for each layer by batch gradient descent [28]. The cost function of a single sample point is as follows:

$$J(W, b; x, y) = \frac{1}{2} ||h_{W,b}(x) - y||^2 \tag{5}$$

The cost function of the entire dataset with m samples is:

$$J(W, b) = [\frac{1}{m} \sum_{i=1}^{m} J(W, b; x^{(i)}, y^{(i)})] + \frac{\lambda}{2} \sum_{l=1}^{n_l - 1} \sum_{i=1}^{s_l} \sum_{j=1}^{s_{l+1}} (W_{ji}^{(l)})^2 \tag{6}$$

Our objective is to optimize W and b such that the cost function is minimized. The batch gradient descent first initializes all parameter $W_{ji}^{(l)}$ and $b_i^{(l)}$ to a random number near zero.

The gradients are obtained by computing the partial derivative of the cost function with respect to $W_{ji}^{(l)}$ and $b_i^{(l)}$:

$$\frac{\partial}{\partial W_{ji}^{(l)}} J(W,b) = \left[\frac{1}{m}\sum_{i=1}^{m}\frac{\partial}{\partial W_{ji}^{(l)}} J(W,b;x^{(i)},y^{(i)})\right] + \lambda W_{ji}^{(l)} \tag{7}$$

$$\frac{\partial}{\partial b_i^{(l)}} J(W,b) = \frac{1}{m}\sum_{i=1}^{m}\frac{\partial}{\partial b_i^{(l)}} J(W,b;x^{(i)},y^{(i)}) \tag{8}$$

Then the parameters are obtained by iteratively minimizing the following functions:

$$W_{ji}^{(l)} = W_{ji}^{(l)} - \alpha\frac{\partial}{\partial W_{ji}^{(l)}} J(W,b) \tag{9}$$

$$b_i^{(l)} = b_i^{(l)} - \alpha\frac{\partial}{\partial b_i^{(l)}} J(W,b) \tag{10}$$

where α is the learning rate, which should be elaborately chosen because the learning rate affects the algorithm's convergence speed, or even leads to divergence.

C. Sparse auto-encoder

The architecture of an auto-encoder is shown in Figure 5, in which a three layer neural network is presented. The basic idea of the auto-encoder is that by enforcing the output values to be equal to the input, the auto-encoder can learning its weight all by itself. Thus, it is an unsupervised feature learning method.

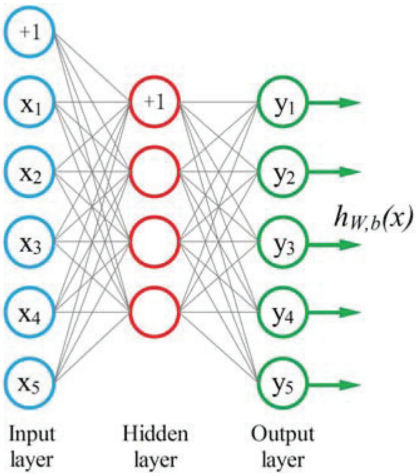

Fig. 5. Basic architecture of sparse auto-encoder. This network learns an alternative representation of the input data by transforming the input data from a feature space to another feature space while imposing constraints that the outputs must be equal to the inputs.

Given a set of training examples without labels, we first perform forward propagation to get the output $h_{W,b}(x)$, hence the error is obtained $\delta(x) = x - h_{W,b}(x)$. We back propagate the error to the input as we optimize the weight $W = \{W^{(1)}, W^{(2)}\}$ according to the gradients in order to minimize the error. We perform the forward-backward propagation iteratively until the weights converge.

This type of network consists of only three layers, two of which are of the same size (the input layer and the output layer have the same number of neurons). The only difference is in the hidden layer. Different number of neurons in the hidden layer can lead to different representations to be learned from the training examples. We can put sparsity constraints on the hidden layer. The sparsity constraints imply that the minimum number of neurons in the hidden layer are activated. The average activation of hidden neuron j is:

$$\hat{\rho}_j = \frac{1}{m}\sum_{i=1}^{m}\left[a_j^{(2)}(x^{(i)})\right] \tag{11}$$

where $a_j^{(2)}(x^{(i)})$ denotes the activation of hidden neuron j. If the average activation of the hidden neuron is enforced to be a small value close to zero, $\hat{\rho}_j = \rho$, hence ρ is called the sparsity parameter. The sparsity constraint is achieved by adding a penalty term in the energy function to penalize $\hat{\rho}_j$ deviating significantly from ρ. The Kullback-Leibler (KL) distance [29] is a desirable metric for this penalty.

$$\sum_{j=1}^{s_2} KL(\rho||\hat{\rho}_j) \tag{12}$$

Hence, the cost function with sparsity constraint can be explicitly formulated as follows:

$$J_s(W,b) = J(W,b) + \beta\sum_{j=1}^{s_2} KL(\rho||\hat{\rho}_j) \tag{13}$$

where $J(W,b)$ is defined in equation 6, and β is the tradeoff between sparsity penalty term and the cost function.

In fact, sparse auto-encoder learns another representation of the input data by transforming the input to the hidden layer. The dimension of the projected feature space depends on the number of hidden neurons. The overall structure of the proposed neural network shown in Figure 4 can be accomplished by stacking multiple sparse auto-encoders together. Let the output of layer i be the input to layer j. Thus the weight $W^{(i)}$ can be learned in an unsupervised way.

D. Softmax classification

Analyzing the data from web can be regarded as the problem of categorization. The number of labels depends on the problem. Generally speaking, most problems we encountered are the multi-label categorization. The Softmax classification model is an efficient classifier that can tackle this problem. This model is the generalization of logistic classification, which is used to decide whether a sample belongs to one of two classes. The softmax classification degrades to logistic classification when there are only two labels. Different from the sparse auto-encoder, softmax classification is accomplished in a supervised way. The softmax classifier can be used to categorize the features extracted by the deep neural network as shown in Figure 6.

Given a training set with m training examples $\{(x^{(i)}, y^{(i)}), ..., (x^{(m)}, y^{(m)})\}$, the parameter θ can be

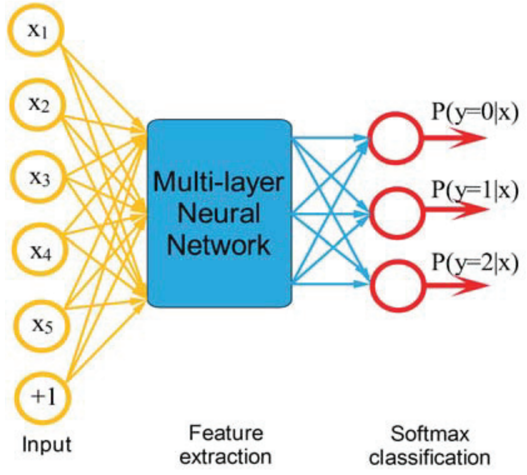

Fig. 6. Softmax classification system consists of three parts. The input data are fed into the multi-layer neural network, followed by the softmax classifier.

vectorized in the following format:

$$\theta = \left[\theta_1^T \theta_2^T \dots \theta_k^T\right]^T \tag{14}$$

The cost function for the training set is:

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{k}\mathbf{1}\{y^{(i)} = j\} log \frac{e^{\theta_j^T x^{(i)}}}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}}\right] \tag{15}$$

where $\mathbf{1}(\cdot)$ is the indicator function, meaning that it is 1 if $(\cdot)$ holds true and 0 if $(\cdot)$ do not. The gradient is as follows:

$$\frac{\partial}{\partial\theta_j}J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[x^{(i)}(\mathbf{1}\{y^{(i)} = j\} - p(y^{(i)} = j|x^{(i)};\theta))\right] \tag{16}$$

where p is the output of softmax classifier:

$$p(y^{(i)} = j|x^{(i)};\theta)) = \frac{e^{\theta_j^T x^{(i)}}}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}} \tag{17}$$

Finally, we can optimize θ by iteratively updating $\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta)$ for $j = 1, ..., k$, where α is the learning rate.

E. Training strategy

The supervised learning and unsupervised learning are two major learning strategies in machine learning. The proposed method encounters both of them, while they are suited for different situations. The supervised way features training the model parameter with the help of groundtruth, while the unsupervised way features learning the model parameter without the help of groundtruth. The supervised learning is often used in training the classifier or the entire neural network. The unsupervised learning is used in learning the weights of sparse auto-encoder. Hence, to choose the appropriate learning strategy depends on the groundtruth label. In our case, we employ the supervised learning for training softmax classifier and the unsupervised learning for training sparse auto-encoder.

IV. Conclusion and Future Work

Nowadays, advances in social media have reduced the cost of the information generation and delivery. A great amount of information is being made every minute, known as the information exploration. The information is organized in a chaotic order. Knowledge discovery in databases is the technique that mines for structured patterns in the raw data. This paper first introduces the development of KDD and the overall system, and then proposes a deep neural network for feature extraction and the softmax classifier for feature extraction. Finally, the different training strategies are discussed. Future work may focus on developing a more effective neural network architecture in order to extract features with more discriminative power. The choice of proper pre-processing algorithm has a great impact on final result, which will also be discussed in the future work.

Acknowledgement

This research is funded by the National High Technology Research and Development Program of China (863 Program) under Grant No. 2013AA01A607 and the Twelfth Five-Year Plan for Xinjiang Manufacturing Information Technology Demonstration Project under Grant No. 201130110-3.

References

[1] J. Vashishtha, D. Kumar, and S. Ratnoo, “Revisiting interestingness measures for knowledge discovery in databases,” in *Advanced Computing & Communication Technologies (ACCT), 2012 Second International Conference on*. IEEE, 2012, pp. 72–78.

[2] M. H. Saraee and B. Theodoulidis, “Knowledge discovery in temporal databases,” 1995.

[3] J. Steele, J. McDonald, and C. D’Arcy, “Knowledge discovery in databases: applications in the electrical power engineering domain,” 1997.

[4] H. Behja, E.-M. Zemmouri, and A. Marzak, “Viewpoint-based annotations for knowledge discovery in databases,” in *Machine and Web Intelligence (ICMWI), 2010 International Conference on*. IEEE, 2010, pp. 320–323.

[5] X. Wang, “Intelligent quality management using knowledge discovery in databases,” in *Computational Intelligence and Software Engineering, 2009. CiSE 2009. International Conference on*. IEEE, 2009, pp. 1–4.

[6] A. Appice, A. Ciampi, A. Lanza, D. Malerba, A. Rapolla, and L. Vetturi, “Geographic knowledge discovery in ingens: An inductive database perspective,” in *Data Mining Workshops, 2008. ICDMW’08. IEEE International Conference on*. IEEE, 2008, pp. 326–331.

[7] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, “Face recognition: A convolutional neural-network approach,” *Neural Networks, IEEE Transactions on*, vol. 8, no. 1, pp. 98–113, 1997.

[8] Y. Zhang and H. Zhang, “Image clustering based on sift-affinity propagation,” in *Fuzzy Systems and Knowledge Discovery (FSKD), 2014 11th International Conference on*. IEEE, 2014, pp. 358–362.

[9] H. Deng, G. Stathopoulos, and C. Y. Suen, “Applying error-correcting output coding to enhance convolutional neural network for target detection and pattern recognition,” in *Pattern Recognition (ICPR), 2010 20th International Conference on*. IEEE, 2010, pp. 4291–4294.

[10] Y. Chen, Z. Lin, X. Zhao, G. Wang, and Y. Gu, “Deep learning-based classification of hyperspectral data,” 2014.

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

[12] 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.

[13] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 7, pp. 971–987, 2002.

[14] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*. IEEE, 2003, pp. 1470–1477.

[15] J. C. van Gemert, J.-M. Geusebroek, C. J. Veenman, and A. W. Smeulders, "Kernel codebooks for scene categorization," in *Computer Vision–ECCV 2008*. Springer, 2008, pp. 696–709.

[16] 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.

[17] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, "Locality-constrained linear coding for image classification," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 3360–3367.

[18] Y. Huang, K. Huang, Y. Yu, and T. Tan, "Salient coding for image classification," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011, pp. 1753–1760.

[19] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *Computer Vision–ECCV 2010*. Springer, 2010, pp. 143–156.

[20] X. Zhou, K. Yu, T. Zhang, and T. S. Huang, "Image classification using super-vector coding of local image descriptors," in *Computer Vision–ECCV 2010*. Springer, 2010, pp. 141–154.

[21] L. Fei-Fei and P. Perona, "A bayesian hierarchical model for learning natural scene categories," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 2. IEEE, 2005, pp. 524–531.

[22] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 2. IEEE, 2006, pp. 2169–2178.

[23] K. Grauman and T. Darrell, "The pyramid match kernel: Discriminative classification with sets of image features," in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 2. IEEE, 2005, pp. 1458–1465.

[24] 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.

[25] H. Bourlard and Y. Kamp, "Auto-association by multilayer perceptrons and singular value decomposition," *Biological cybernetics*, vol. 59, no. 4-5, pp. 291–294, 1988.

[26] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 1096–1103.

[27] A. Coates, A. Y. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *International Conference on Artificial Intelligence and Statistics*, 2011, pp. 215–223.

[28] T. Nakama, "Theoretical analysis of batch and on-line training for gradient descent learning in neural networks," *Neurocomputing*, vol. 73, no. 1, pp. 151–159, 2009.

[29] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, pp. 79–86, 1951.