Journal of Network Intelligence ©2016 ISSN 2414-8105(Online)
Taiwan Ubiquitous Information Volume **1**, Number **3**, July **2016**

Overview of Deep Kernel Learning Based Techniques and Applications

Xiu-Yuan Chen, Xi-Yuan Peng, Jun-Bao Li*, Yu Peng

Department of Automatic Test and Control
Harbin Institute of Technology, China
Email: junbaolihit@126.com
*Corresponding author

Received April 2016; revised July 2016

Abstract. *Machine learning, especially neural network algorithm, gets more attentions in the last decades. With the in-depth study of intelligence algorithm and network structure, machine learning is widely used in many aspects, such as data mining, computer vision, data recognition and classification. Because of the nonlinear and complex characters of target data, the researches above need fairly accurate characteristics space extracted from the data space. This process relies on machine learning to perform better, since artificial rules cant obtain the most efficient features. Researchers combine kernel methods and deep neural network to keep their advantages and make up their defects, then apply deep kernel learning to improve the algorithm performance in application. In this paper, we present an overview of the research progress of deep kernel learning and its applications. We introduce the basic theories and their fusion which form several deep kernel learning structures to enhance algorithm properties and performance in practice.*
Keywords: Kernel method, Deep neural network, Deep kernel learning.

1. Introduction.

The essence of machine learning is to find out the accuracy mapping from the sample space to the feature space, and the following applications rely on the linearly separable of the features, which means feature extraction is one of the most important things in learning. One major direction to improve the performance of a feature extractor is to identify flexible learning models that can discover the nature of various types of datasets, especially nonlinear datasets, then kernel methods are proposed following this direction to bring nonlinear ability to linear methods[1]. Kernel methods work by mapping the data from the input space into a high-dimensional feature space, and then building linear algorithms in the feature space to implement nonlinear counterparts in the input space. The mapping is determined by specifying a kernel function, which computes the inner product between each pair of data points in the feature space[2]. Kernel-based learning algorithms, which include SVMs, kernel PCA, and Gaussian Processes, have become well-established tools that are useful in a variety of contexts, including discriminative classification, regression, density estimation, and clustering[3]. However, their generalization ability is limited by the choice of kernel function. That means choosing the appropriate kernel function has a crucial effect on the performance of the kernel methods and the applications.

Multiple kernel, optimization via learning procedure[4] and data-dependent kernels[5] are the primary approaches for optimization. Researchers attempt to solve the limitation of simple kernel methods with MKL (multiple kernel learning). It aims at learning a linear (or convex) combination of a set of predefined kernels in order to determine the best target kernel for the application. During the last decade, MKL has been widely investigated, and many algorithms and techniques were proposed in order to improve the efficiency of this new method[6]. Although these approaches mentioned above enhance the learning ability of kernel methods, they are still shallow and local methods that essentially limit the ability of the algorithm to adequately learn the nature of highly variable datasets [7]. In order to solve this problem, deep learning architecture of neural networks was proposed by Hinton first in 2006[8]. The deep learning architecture has powerful ability to learn functions or distributions from the datasets automatically[9]. Compared to kernel methods, it is deep and nonlocal, and thus exhibits more powerful ability to learn the nature of highly variable datasets from fewer samples. On the other hand, the unsupervised pre-training phase involved in the deep learning architecture regularizes the learning procedure to produce better generalization performance. The results of recent researches show that deep architecture is much more promising compared to the shallow models. Deep architecture can be introduced into many learning algorithms to improve their performance and extend their application range. Brahma and Dapeng Wu[10] have proved the feasibility of deep architecture for learning algorithms and Fabio Anselmi[11] demonstrates that deep convolutional networks are Hierarchical Kernel Machines.

Therefore, deep kernel learning, a new area of machine learning, was proposed in order to apply the idea of deep learning in order to improve the kernel learning task and seek for some new optimal methods. Deep kernel learning combines the structural properties of deep learning architecture with the non-parametric flexibility of kernel method and our paper is an overview of deep kernel learning based techniques and applications. The rest of this paper is organized as follows: In section 2, we give more details for the related work, and we show the basic theories of how deep kernel learning can be constructed in section 3. Section 4 exhibits the combination of kernel methods and deep architectures through their applications and extensions. The final section 5 summarizes this paper and gives the outlook of future work in deep kernel learning.

2. **Related work.**

To achieve the research of deep kernel learning, researchers mainly proposed ideas and methods in two ways. The first direction is to combine the kernel function and deep architecture and optimize with learning algorithms. Hinton and Salakhutdinov[12] combine deep belief networks (DBNs) with Gaussian processes, showing improved performance over standard GPs with RBF kernels, in the context of semi-supervised learning. Their model is heavily relying on unsupervised pretraining of DBNs, with the GP component unable to scale beyond a few thousand training points. Calandra et al.[13] combine a feedforward neural network transformation with a Gaussian process, showing an ability to learn sharp discontinuities, and the resulting model can only scale to at most a few thousand data points. Huang et al.[14] pursue an unsupervised pre-training procedure using deep auto-encoders, showing improved performance over GPs using standard kernels. Wilson et al.[15] combine deep feedforward and convolutional architectures with spectral mixture covariances [16], inducing points, and local kernel interpolation[17]. They achieve scalability while retaining non-parametric model structure by leveraging the very recent KISS-GP approach and extensions[18] for efficiently representing kernel functions, to produce scalable deep kernels.

In order to improve the feature extraction ability, WK Wang and M Sun propose a new feature extraction method called regularized deep Fisher mapping (RDFM)[19] , which finds a mapping from the sample space to the feature space using a deep neural network to enhance the separability of features according to the Fisher criterion. In recent years, the Convolutional Neural Network is the most popular in deep learning structure. In some variants of CNN, SVMs have been used in the final layer to generate the output by utilizing its wide-margin classification capabilities. Yang et al. combine convolutional networks, with parameters pre-trained on ImageNet, with a scalable Fastfood[20] expansion for the RBF kernel applied to the final layer[21].

The second direction is to extend kernel methods to deep kernel architecture. Researchers propose some deep architecture based on kernel functions to improve the shallow learning machines. The first kernel method using deep architecture was introduced by Cho and Saul[22]. Their main idea consists on iteratively repeating the inputs mapping L times and a new family of kernel functions, called arccosine kernels, was proposed; it imitates the computational process of the deep neural network. Some emerging works have applied the idea of deep learning in order to improve the MKL methods. Since the deep architecture shows high performance over the shallow one, several studies extended the single layer of multiple kernels by learning multilayers of multiple kernels, and this framework is usually called Multilayer Multiple Kernel Learning (MLMKL) which represents a new active research area for kernel methods. These new methods tend to learn a deep architecture by exploring the combinations of multiple weighted kernels in a multilayer structure and the main goal is to learn the optimal kernel combinations and the decision function. The base kernels, in each layer, are combined and used as input to the base kernels in the immediate layer. Zhuang et al.[23] propose to optimize the system based on the dual objective function. The authors were able to optimize an MLMKL of only two layers, where the second layer is composed of a single RBF kernel. Strobl and Visweswaran [24] optimized multiple layers of kernels over an estimation of the leave-one-out error method.

Another widely used application of kernel method called KPCA perform efficiently in the proposed architecture called Feedforward kernel neural networks(FKNN). The number of the layers of a FKNN network can be increased in a layer-by-layer way and a KPCA behaves as a data transformation method in each hidden layer[25]. Wang et al. proposed a KPCA learning architecture based FKNN networks[26]. Chao and Saul implemented deep learning by using a special multi-layer kernel machines in[22]. Mitchell et al. [27] realized a deep structure learning by using multi-layer linear PCAs and pointed out that their deep structure learning framework can be used to describe the conventional Convolutional Networks and Deep Belief Networks. With the research of the structure in depth, except for linear PCAs more nonlinear KPCAs and more kernel functions can be involved and more error functions can be considered within this framework.

3. **Algorithm and theory.**

In the research of Deep Kernel Learning, the algorithms and theories are applied to achieve three functions: feature extraction, classifier and parameter optimization. Deep learning structure is mainly used to improve the extraction performance and kernel applications are mainly used as classifier. When apply the deep learning and kernel applications together, the matrix transferred between them should be regulated and the parameters need to be optimized in appropriate ways. In this section, we introduce the basic algorithms and theories used in deep kernel learning.

3.1. **Deep learning structure.**

Deep neural networks (DNNs) are argued to have a greater capacity to recognize a larger variety of visual patterns than shallow models, because they are considered biologically plausible. Deep learning methods theoretically exhibit powerful learning ability to discover the nature of the dataset from samples. Erhan et al. [28] draws an important conclusion from extensive experiments of deep learning algorithms: the unsupervised pretraining phases of the deep learning methods act as regularizers by restricting the parameters to a relatively small region of parameter space that corresponds to capturing structure in the input distribution.

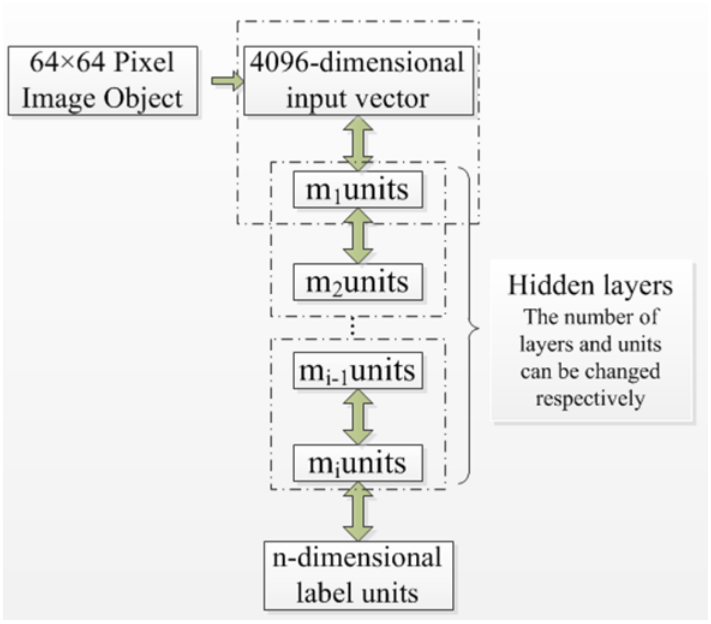
64×64 Pixel Image Object
4096-dimensional input vector
$m_1$units
$m_2$units
m_{i-1}units
$m_i$units
n-dimensional label units
Hidden layers
The number of layers and units can be changed respectively

Figure 1. The construction of Deep Belief Network

3.1.1. *DBN.* The Deep Belief Network is composed of several hidden layers of hidden units with connections between the layers but not between units within each layer. When training the object in an unsupervised way, the DBN learns to reconstruct the object as input vectors, and the hidden layers act as feature detectors on inputs, so that the DBN can be used to perform classification [29]. Figure 1 shows a common construction of the Deep Belief Network.

3.1.2. *CNN.* CNN is a multi-layer neural network and a common structure is shown in Fig. X. It is an important method widely used in image recognition. Each layer of CNN consists of many two-dimensional planes, and each plane consists of many single neurons [30]. Each neuron of those maps defines their receptive fields [31], and the neurons only accept the signal from local receptive fields. In general, layer C is feature extraction layer, and each input neuron of C is connected to the local receptive field in the input and extracts its local features and then determines the location relationship with other feature space by those local features. Layer S is feature mapping layer that has the advantage of position invariance. Each feature mapping is a plane, and all neurons' weights in the plane are equal, thus reducing the number of free parameters and reducing the complexity of network parameters. Each feature extraction layer (layer C) will follow a subsampling layer (layer C) which constitutes the two feature extraction, so the network has good distortion tolerance [32].

The training of CNN is supervised, and it can be divided into three steps:
Step 1: Through forward propagation to compute the output of each layer;
Step 2: By the above network output, BP algorithm is used to seek the derivative of error on the network weights;
Step 3: Through the weight updating method to update the weights.

CNN is a kind of artificial neural network, and it is good at mining datas local features. CNN adopts the weight sharing technique on one feature map resulting in reducing network models complexity and decreasing the number of weights. These advantages make CNNs applied to many fields in pattern recognition [33,34]. After years of research, CNNs

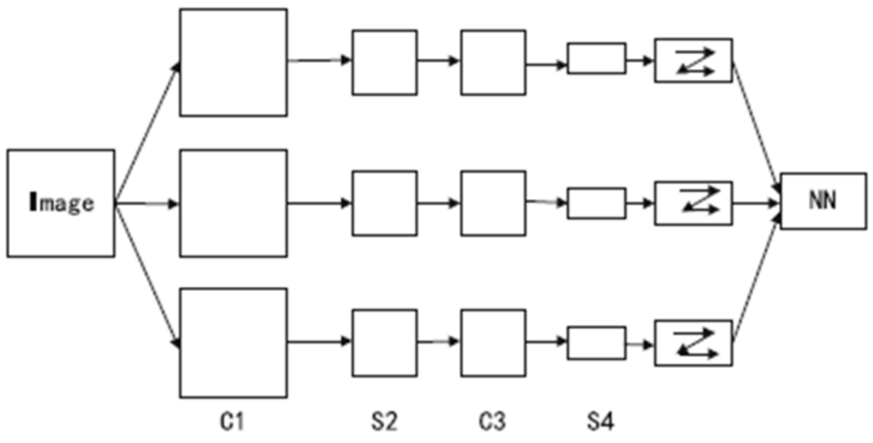

FIGURE 2. Architecture of CNN

successfully applied in face detection, document analysis, speech detection, license plate detection, etc. The basic architecture of CNN is shown in Figure 2.

However, Current deep learning algorithms always do not involve unsupervised regularizers in the fine-tuning phase, but use early stopping strategies and tiny learning rates to "avoid unduly perturbing the weights found by the pre-training". Besides, the performance of CNN relies on the large amount of training dataset which contains millions of images.

3.2. **Kernel application.** Kernel methods have been particularly successful on a variety of sample sizes because they can enable a classifier to learn a complex decision boundary with only a few parameters by projecting the data onto a high-dimensional reproducing kernel Hilbert space. The basis of kernel methods is kernel function and the most common applications are SVM and multiple kernel learning.

3.2.1. *Support vector machine.* Support vector machines are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. The initial SVMs are used to solve linear problems and these machines can perform nonlinear tasks with the help of kernel. Classification of images and hand-written characters recognition can be performed using SVM. SVMs produce lower peak recognition rates, but could be trained to reach close to those peaks using much less training data as compared to deep learning techniques[35].

3.2.2. *Multiple Kernel Learning.* Multiple kernel learning consists of combining a set of already defined base kernels so as to learn the optimal kernel. It exhibits its strength of learning multiple kernels and its capability of combining heterogeneous data [36]. The basic structure of MKL is shown in Fig. 3.

During the last decade, MKL has been actively investigated, in which lots of algorithms and techniques were proposed in order to improve the efficiency of the MKL method [37, 38, 39]. In [40], the authors solved MKL network using the sequential minimization optimization (SMO) algorithm, while in [41], the authors used semi-infinite linear programming (SILP) method for solving large-scale MKL problems. The experimental results have shown that these methods often achieve worse prediction results than the simple average of the base kernels [42]. Although MKL approach has been widely studied, its architecture is still "shallow", since it consists of a simple (linear/convex) combination of multiple kernels. Indeed, the optimal kernel is not rich enough to solve complicated applications.

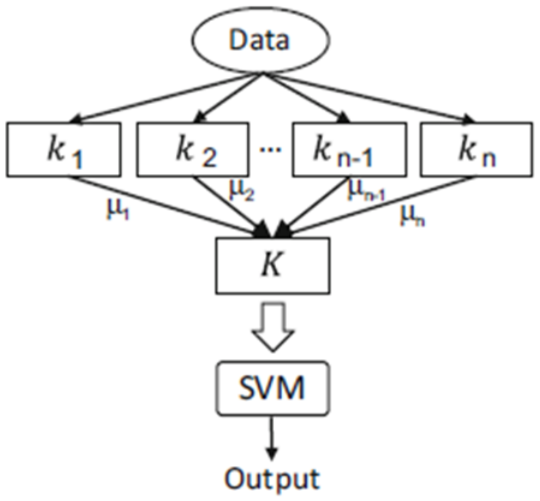

FIGURE 3. Multiple kernel method architecture

4. **Applications and extensions.** The applications and extensions of deep kernel learning is mainly achieved in two directions. One is the combination of deep network architecture and kernel methods, and the other is to structure the deep kernel learning with shallow kernel algorithms. Such deep kernel learning framework aims at learning useful features by stacking successive layers and each layer is issued from a nonlinear projection of output of the previous layer using kernel method. In this section, we introduce several deep kernel methods which have been proved and tested by a large amount of datasets and experiments.

4.1. **DBN and Kernel Methods.** In general, there are two steps in deep learning. First, features of each layer are pre-trained with layer-wise unsupervised learning, and these pre-trained features are then fine-tuned throughout whole layers by supervised learning. Like the multi-layer perceptron in [43], the whole-layer fine-tuning only utilizes global level features, which means features from different hierarchies are intractable for classification problems. Moreover, the study[44] suggests that globally trained features from each layer are concatenated into single vector representations which is classified by SVM. However, those simple concatenations have limitations as well in mathematical sense.

Multiple Kernel Learning (MKL) is proposed to optimize the combinations of kernels via finding the suitable θ and simultaneously update the discriminative weights u. As a result, MKL achieves the optimal combination of various kernels which measure the similarity of input in different manner or originate from diverse sources. In short, multiple features are able to be concatenated via MKL method. In addition, MKL can analyze the best kernel for the given machine learning problem.

In order to combine hierarchical feature representations learned and estimated under DBN model, the authors in [45] applied MKL in their study and choose Ultra-Fast Online Multiple Kernel learning algorithms [46] that use the specific formation of regularization norm. By selecting the regularization function, UFO MKL model can achieve optimal convergence rate only depending on the logarithm of the number of kernels. The combinations of hierarchical feature representations are optimized by UFO MKL algorithms and each kernel function of MKL is replaced by each level of features from DBN.

4.2. **Deep kernel learning and Gaussian processes.** In paper [47], the authors show that the proposed model outperforms state of the art stand-alone deep learning architectures and Gaussian processes with advanced kernel learning procedures on a wide range of

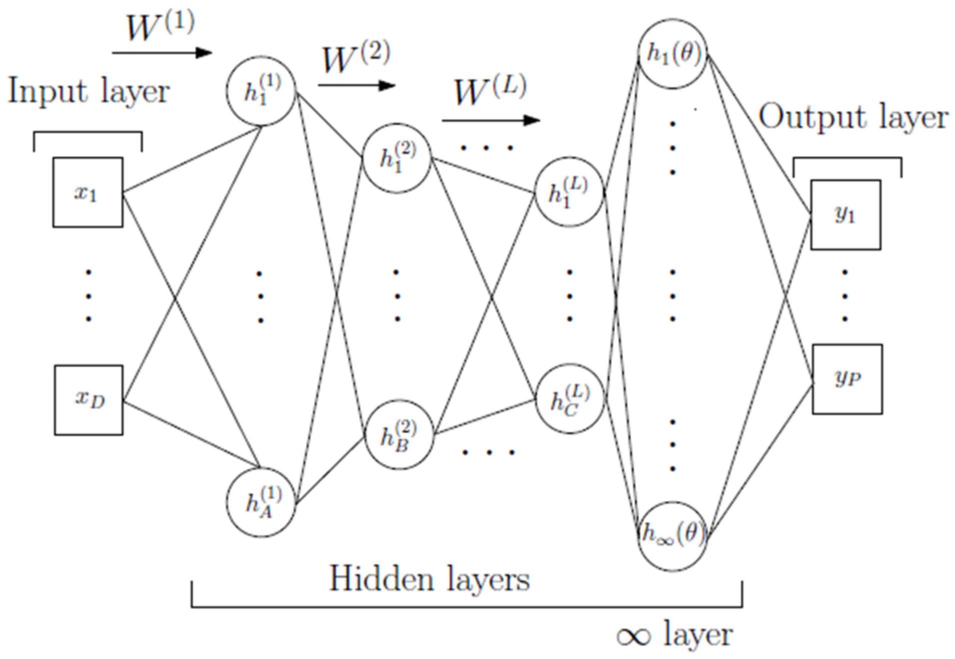

FIGURE 4. Architecture of deep kernel learning

datasets, demonstrating its practical significance. Specifically, starting from a base kernel $k(x_i, x_j\,|\theta)$ with hyperparameters θ, they transform the inputs x as

$$k(\mathrm{x}_i, \mathrm{x}_j|\theta) \rightarrow k(g(\mathrm{x}_i, \mathrm{w}), g(\mathrm{x}_j, \mathrm{w})|\theta, \mathrm{w}) \tag{1}$$

where *g(x, w)* is a non-linear mapping given by a deep architecture, so that the deep neural network and kernel learning are combined and parametrized by weights w. The popular RBF kernel is a sensible choice of base kernel, and researchers also propose to use spectral mixture base kernel [48] for added flexibility:

$$k_{\mathrm{SM}}(\mathrm{x}, \mathrm{x'}|\theta) = \sum_{q=1}^{Q} a_q \frac{\left|\sum_q\right|^{\frac{1}{2}}}{(2\pi)^{\frac{D}{2}}} \exp(-\frac{1}{2}\left\|\sum\nolimits_q^{\frac{1}{2}} (\mathrm{x} - \mathrm{x'})\right\|^2) \cos\langle \mathrm{x} - \mathrm{x'},\ 2\pi\mu_q\rangle \tag{2}$$

The spectral mixture (SM) kernel, which forms an expressive basis for all stationary covariance functions, can discover quasi-periodic stationary structure with an interpretable and succinct representation, while the deep learning transformation $g(x, w)$ captures non-stationary and hierarchical structure.

The deep kernel of Eq. 1 is used as the covariance function of a Gaussian process to model data $D = \{x_i, y_i\}_{i=1}^{n}$. Conditioned on all kernel hyperparameters, the model is interpreted as applying a Gaussian process with base kernel to the final hidden layer of a deep network and the model is shown in Figure 4.

This deep kernel learning is a Gaussian process with a deep kernel maps D dimensional inputs x through L parametric hidden layers followed by a hidden layer with an infinite number of basic functions. All the deep kernel hyperparameters $\gamma = \{w, \theta\}$ are jointly learned which include w, the weights of the network, and θ the parameters of the base kernel, by maximizing the log marginal likelihood L of the Gaussian process.

4.3. **Deep Fisher Kernel Learning.** In deep fisher kernel learning, the researchers propose a new feature extraction method called regularized deep Fisher mapping (RDFM), which learns an explicit mapping from the sample space to the feature space using a deep neural network to enhance the separability of features according to the Fisher criterion. So the deep learning architecture is used to enhance the learning ability of FDA and optimize the Fisher criterion. For example, [49] used multilayer perceptrons to optimize the Fisher criterion and [50, 51] also used deep neural networks to optimize the Fisher criterion. However, Cruz's method is not regularized by any unsupervised information and Stuhlsatza's method does not regularize the fine-tuning phase.

In [52] the researchers use the regularized deep learning architecture to extract effective features for better class separability measured by the Fisher criterion and analyze the effectiveness of unsupervised regularizers applied in both the pretraining phase and the fine-tuning phase. The learning procedure of RDFM, which follows the deep learning architecture in three phases, is elaborated in the following sections.

1) Unsupervised Pretraining: RBM networks are used to pretrain the network. Therefore, when an RBM is trained, it is unrolled to form a small auto-encoder with only one hidden layer and optimized with objective R for several steps. This trained small auto-encoder is denoted as unsupervised belief network (UBN). Both RBM and UBN strategies are tested and analyzed in this phase.

2) Unrolling: In the unrolling phase, the layers of neurons in the RBMs or UBNs trained in the previous pre-training phase are united and rearranged to form one large deep auto-encoder network, where the hidden layers of RBMs or UBNs are connected sequentially to initialize the encoder part of RDFN, and the output layers of UBNs or the visual layers of RBMs are connected sequentially to initialize the decoder part of RDFN.

3) Supervised Fine-Tuning: In this step, the network is fine-tuned to extract effective features for classification by optimizing objective. The conjugate gradients method is adopted for objective optimization, in which the gradients $\partial E/\partial \mathrm{W}$ are computed by backpropagation [53].

4.4. **CNN and kernel methods.** Besides the deep belief network, another deep network architecture called convolutional neural network (CNN) which is famous for its outstanding performance in image identification and classification. For wider applicability, CNN techniques need to be developed to train faster on a smaller database of training samples, particularly when sufficient training samples of target classes are not available. This kernel application is a Siamese CNN with appropriate post-processing of the network output to produce gram matrices corresponding to a reproducing kernel Hilbert space that can be used in a support vector machine (SVM) for classification.

The authors of [54] proposed a novel model of convolutional extreme learning machine with kernel (CKELM) which combines the CNN, ELM, and Kernel together to improve the performance of learning machine. Extreme learning machine (ELM) proposed by Huang et al. is a learning algorithm for single-hidden layer feedforward neural networks (SLFNs). Extreme learning machine with kernel (KELM) is an application of extreme learning machine which has the advantage of fast learning speed and high efficiency, as well as the deficient in feature extraction for its single-hidden layer structure. In order to solve the problem, The CKELM was proposed with convolutional layers and subsampling layers alternately added to hidden layer of the original KELM to extract features and classify. Random weights are used to adjust parameters instead of the gradient algorithm. The architecture of ELM and CKELM are shown below.

This new model named CKELM improves the accuracy when compared with other ELM-based methods and the time is far less than other deep learning methods. It is worth seeking for more efficient and faster methods.

Unlike the typical CNN which computes class probability of a single input image, the proposed kernel CNN in paper [55] takes two images as input and is trained to compute

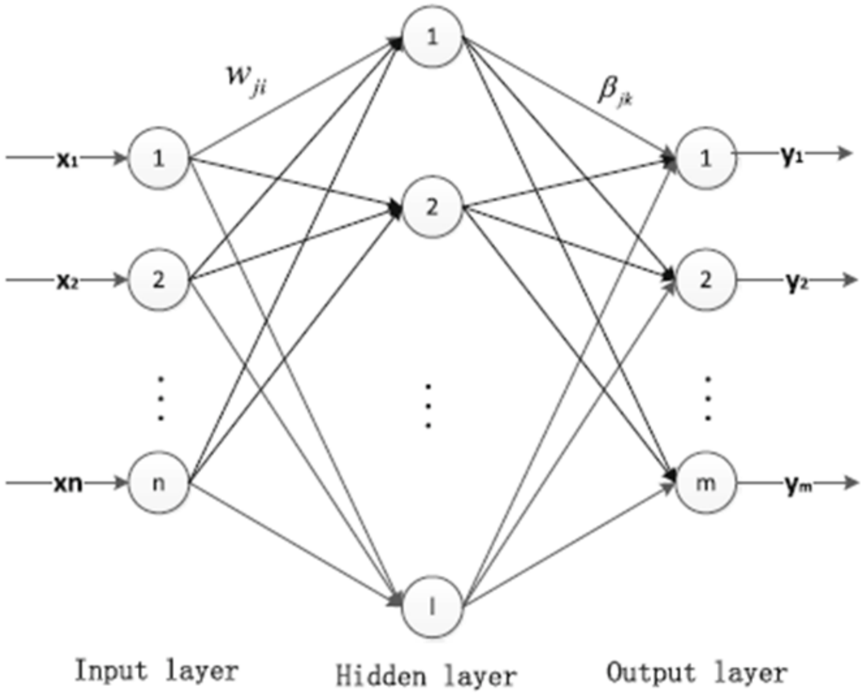

FIGURE 5. Architecture of ELM

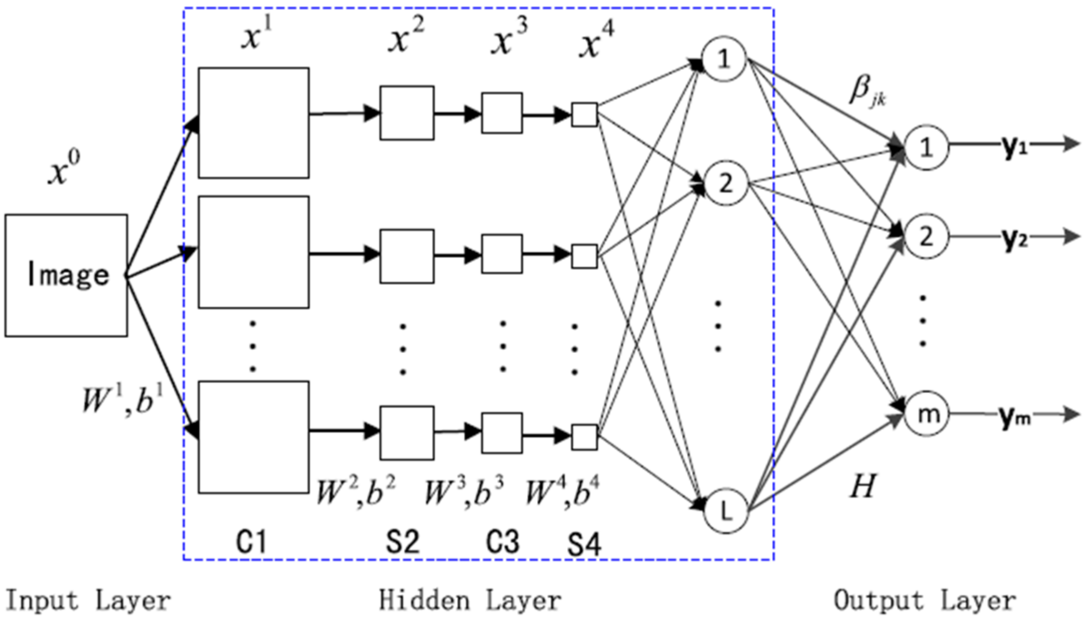

FIGURE 6. Architecture of CKELM

similarity between them and obtain the probability they belong to the same class, no matter what that class is. And the architecture is divided into three parts:

- *Training the NN with transfer learning*: The neural network was trained in two different phases to facilitate two levels of transfer learning. One for the convolutional layers and the other for fully connected layers. In the first level of transfer learning, one can copy the convolutional weights of a CNN trained to classify samples of non-target classes into DESKs neural network for target classes. In the next level of

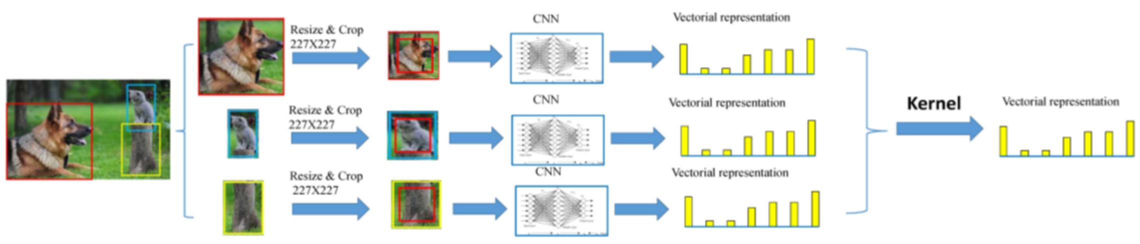

Figure 7. The proposed kernelized convolutional neural network algorithm

transfer learning, one can train the fully connected layers on pairs of images from non-target classes.

- *From NN to kernel*: The neural network output is not guaranteed to satisfy the Mercers theorem. Therefore, before training an SVM, its necessary to make the output symmetric and the kernel matrix positive semi-definite.
- *SVM training and testing*: The kernel matrix thus obtained was used in an SVM to learn the indices and weights of the training samples that serve as support vectors. The SVMs only hyperparameter (the slack penalty) was set using cross-validation. The trained SVM was used for classification of testing samples using the gram matrix of testing and training samples.

In paper [56], the authors extract CNN features on the detected object-like patches and aggregate these patch-level CNN features to form a vectorial representation with the Fisher vector model. The CNN model is adopted to transform the image patches into its vectorial representationes and after that the researchers adopt the separable kernel methods to aggregate the representations together to represent the images. One classic separable kernel is the Fisher kernel which models the joint probability distribution of a set of features.

This KCNN method is proposed to describe the content of complex images, and with this method, a more robust vectorial representation is obtained. Besides the CNN structure implemented in Caffe library, there are also some other emerging CNN structures, which can be integrated to improve the performance.

In paper [57], the CNN is used as a trainable feature extractor to extract features from the textual data. The CNN was trained using a standard back-propagation procedure. In this way, the last output layer of the CNN is used only for training, but for actual decision-making, it is replaced with much more sophisticated classifiers, namely, with SVM or MKL.

In the last few decades, researchers have developed successful kernel-based systems for a wide range of visual recognition tasks. Those sensibly-designed kernel functions provide an extremely valuable source of prior knowledge, which should be exploited in deep learning. In paper [58], the researchers propose an informative kernel-based regularizer, which makes it possible to train DNNs with prior knowledge about the recognition task. The kernel regularizer is transformed into a loss function represented as a sum of costs by individual examples. This results in a simple multi-task architecture where a number of extra nodes at the output layer are added to fit a set of auxiliary functions automatically constructed from the kernel function. Usually, machine learning is realized by minimizing a loss function.

$$L(\beta,\theta)=\sum_{i=1}^{n} l(y_i,\beta_1^{\mathrm{T}}\phi_i+\beta_0)+\lambda\|\beta_1\|^2 \tag{3}$$

$$L(\alpha,\beta_0,\theta)=\sum_{i=1}^{n} l\left(y_i,\sum_{j=1}^{n}\alpha_j K_{i,j}+\beta_0\right)+\lambda\sum_{i,j=1}^{n}\alpha_i\alpha_j K_{i,j} \tag{4}$$

$$K_{i,j}=\langle\phi(x_i;\theta),\phi(x_j;\theta)\rangle=\phi_i^{\mathrm{T}}\phi_j \tag{5}$$

Function (4) is the result of deriving the equivalence to a kernel system, and the kernel is computed by function (5). The network is provided with some prior knowledge, and this prior knowledge is exploited via imposing a kernel regularization so that the learning problem seeks

$$\min_{\beta,\theta} L(\beta,\theta)+\gamma\Omega(\theta) \tag{6}$$

The described method is applied to train convolutional neural networks (CNNs) for a wide range of visual recognition tasks. Our results show that incorporation of prior knowledge can boost the performance of CNNs by a large margin when the training set is small or the learning problem is difficult. This paper also explains why bother training with kCNN, instead of simply combining two independently trained CNN and SVM systems. The reason is computational speed kCNN pays an extra cost to exploit a kernel matrix in the training phase, but in the prediction phase the system uses CNN alone.

4.5. **Deep Multilayer Multiple Kernel Learning.** Although the limitation of simple kernel methods is solved by MKL, the property is influenced by the choice of base kernels. So deep learning architecture is introduced for its outstanding learning ability to improve MKL. The first attempt of Multilayer Multiple Kernel learning is a Two-Layer Multiple Kernel Learning framework. In paper [59], the authors optimize an MLMKL of only two layers, where the second layer is composed of a single RBF kernel. The two-layer multiple kernel domain as follows

$$K^{(2)}=\left\{k^{(2)}(x_i,x_j;\mu)=\exp\left(\sum_{t=1}^{m}\mu_t k_t^{(1)}(x_i,x_j)\right):\mu\in\mathbb{R}_+^m\right\} \tag{7}$$

Since $k^{(2)}$ is positive semi-definite, the objective is convex over α. Thus it can be solved by standard QP solvers for a fixed μ. The objective function to optimize is show:

$$\min_\mu \max_\alpha \sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j k^{(2)}(x_i,x_j;\mu)+\sum_{t=1}^{m}\mu_t \tag{8}$$

Once solving the above optimization to find the solutions for α and μ, it's straightforward to obtain the final decision function of the Two-Layer MKL machine:

$$f(x;\alpha,\mu)=\sum_{i=1}^{n}\alpha_i y_i k^{(2)}(x_i,x;\mu)+b \tag{9}$$

The authors implement an MLMKL framework in which all base kernels in antecedent layers are combined so as to form new inputs to the base kernels in subsequent layers. Instead of using a single kernel function, a set of functions, organized in a specific structure, map the original data through several layers of kernels, and then, the final kernel is used to learn the decision function of SVM. The structure of this framework is shown in Fig.8.

The resultant kernel K of the multilayer multiple kernel domain is defined as follows:

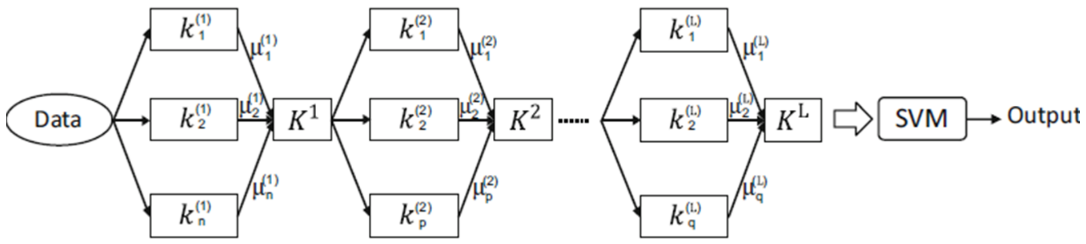

FIGURE 8. Structure of multiple layer multiple kernel framework

$$K^{(l)} = \left\{ K^{(l)}(K^{(l-1)};\mu^{(l)}) = \sum_{k=1}^{M} \mu_k^{(l)} k_k^{(l)}(K^{(l-1)}) \,\middle|\, \mu_k^{(l)} \geq 0,\ k = 1,...,M,\ l = 1,...,L \right\}$$
$$K^{(1)} = K^{(1)}(x, x_i; \mu^{(1)}) = \sum_{k=1}^{M} \mu_k k_k(x, x_i) \tag{10}$$

Where $k_k^{(l)}$ is the kth base kernel at layer l and μ_k^l denotes the weight of the kth kernel at layer l. The final decision function of the proposed framework is defined as:

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i K^{(L)} \left(K^{(L-1)}; \mu^{(L)}\right) + b \tag{11}$$

The backpropagation algorithm and the gradient ascent are applied to obtain the coefficients μ that minimize the true risk of the decision function. The alternating optimization algorithm is developed to learn the decision function and all the network weights simultaneously, and this is done as follows: (1) fix α and solve μ and (2) fix μ and α solve.

The algorithm consists on combining multiple base kernels in antecedent layers to form new inputs to the base kernels in subsequent layers, and the proposed optimization method achieved better performance than the existing algorithms based on either the dual objective function or the estimation of the leave-one-out error. However, some improvements need to be achieved such as combining the effectiveness of the proposed method with the capacity of multiple classifier methods.

4.6. **Deep FKNN network.** In paper [61], the authors first propose an architecture, called FKNN, of feedforward neural networks with infinitely differential kernel based activation functions, which can include a large family of existing feedforward neural networks such as radial basis function (RBF) networks, sigmodial feedforward networks, self-organizing feature map (SOM) networks and wavelet neural networks [62], and hence can meet most practical requirements. The contributions of this work can be highlighted in two main aspects: (1) When the number of the hidden nodes of every hidden layer and the type of the adopted kernel based activation functions are prefixed, different from the common understanding of learning, we prove that when all the hidden layers of such networks are tuningfree and their parameters are randomly assigned and even may be independent of the training data, such networks are universal approximators with probability one. Therefore, the least learning machine (LLM) [63], as a generalized version of the extreme learning machine (ELM) [64], can be further extended into its generalized version in the sense of adopting much more error functions instead of only the MSE function. (2) When the proposed architecture of FKNN networks is constructed in a layer-by-layer way, i.e., the number of the hidden nodes of every hidden layer may be determined after the explicit execution of a KPCA, we can develop FKNNs deep architecture such that its deep learning has strong theoretical guarantee with the enhanced performance for image classification. The detailed framework DLF for a deep FKNN network is described as follow:

- ***Input:*** The given dataset; Number of the hidden layers, with their respective kernel functions used in multi-layer KPCAs, of a deep FKNN network;
- ***Output:*** The parameters of the output layer determined by a LA, and/or the kernel parameter vectors in every hidden layer of a deep FKNN network;
  -*Step1:* Preprocess the given dataset form the input dataset for the deep FKNN network; fix the number of the layers in the FKNN, and the type of a kernel function in each hidden layer of the deep FKNN network;
  -*Step2:* Carry out a full-rank or low-rank KPCA with the chosen kernel function and its initial kernel parameter vector in every hidden layer, in a layer-by-layer way from the input layer to the last hidden layer. If a full-rank KPCA is taken, the number of the hidden nodes in a hidden layer is automatically determined by the full-rank KPCA. Otherwise, fix the chosen top-k principal components as the number of the hidden nodes, in terms of the extracted principal components by a low-rank KPCA.
  -*Step3:* Carry out the chosen learning algorithm LA in the output layer on the transformed dataset after multi-layer KPCAs such that the parameters in the output layer are determined and the kernel parameter vectors in the last hidden layer are optimized.
  -*Step4:* Determine an appropriate kernel parameter vector in every hidden layer by:
  (1) The grid search in every hidden layer in a layer-by-layer way from the last hidden layer to the input layer; Or
  (2) Choose certain optimization method with certain performance criterion (e.g. MSE) and the initial kernel parameter vector to optimize the kernel parameter vector in every hidden layer in a layer-by-layer way from the last hidden layer to the input layer.

4.7. **Arc-cosine kernels mimic neural nets.** The paper [65] introduces a new kernel function called arc-cosine kernel that mimics the computation in large, multilayer neural nets. These kernel functions can be used in shallow architectures, such as support vector machines (SVMs), or in deep kernel-based architectures called multilayer kernel machines. The nth order arc-cosine kernel function via the integral representation is defined as follow:

$$k_n(x,y) = 2\int \mathrm{d}w \frac{e^{-\frac{\|w\|^2}{2}}}{(2\pi)^{d/2}} \Theta(w\cdot x)\Theta(w\cdot y)(w\cdot x)^n(w\cdot y)^n \tag{12}$$

The integral in eq.(12) has a simple, trivial dependence on the magnitudes of the inputs x and y, but a complex, interesting dependence on the angle between them.

$$k_n(x,y) = \frac{1}{\pi}\|x\|^n\|y\|^n J_n(\theta) \tag{13}$$

$$J_n(\theta) = (-1)^n(\sin\theta)^{2n+1}\left(\frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\right)^n\left(\frac{\pi-\theta}{\sin\theta}\right) \tag{14}$$

This new kernel function imitates the computational process of the deep neural network. Specifically, L layer's architecture is the inner product after L feature mapping kernels of the input data:

$$K^{(L)}(x,y) = \Phi^{(L)}(...\Phi^{(1)}(x)) \cdot \Phi^{(L)}(...\Phi^{(1)}(y)) \tag{15}$$

where x and y are the input vectors, $\Phi^{(L)}$ is a feature mapping function applied L times, and $K^{(L)}$ represents the final layer kernel. Unlike the basic kernel function, arc-cosine kernel ensures the performance of kernel function in multilayer architecture. In addition, the proposed multilayer kernel has a static architecture.

5. **Conclusion and future work.** The research of deep kernel learning at the present stage is mainly processed in two directions. One direction is to combine the deep learning structure and kernel applications, which uses the deep architectures for feature extraction and the kernel applications as classifier. This direction is gradual developing from simple combination to fusion deep algorithm. The other direction is to extend kernel methods to deep kernel architecture. During the learning and optimization process, several methods are used to optimize and adjust the parameters.

As future work, the idea is and there are several aspects to improve the performance of this deep kernel learning. Instead of several given kernel widths, the appropriate values should be determined by certain optimization method. Since practical applications sometimes require real-time operation, how to speed up its learning is an interesting topic. The proposed deep kernel network can provide a wide range of feature representations at varying levels of abstraction, how to improve current deep learning framework version so that its learning can synthesize feature representations at varying levels of abstraction is another interesting topic. Another interesting topic is how to improve current deep learning framework so that it can share the common transformed data space for multiple tasks. Although the proposed deep learning frameworks have been justified by experiment results about image classification, their effectiveness should be further witnessed by exploiting their more applications and doing comparative study with the existing deep learning methods.

REFERENCES

[1] J. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, Face recognition using kernel direct discriminant analysis algorithms, *IEEE Trans. Neural Netw.*, vol. 14, no. 1, pp. 117126, 2003.

[2] T. Wang, D. Zhao, S. Tian, An overview of kernel alignment and its applications, *Artificial Intelligence Review*, vol. 43, no. 2, pp. 179-192, 2015.

[3] K. Grauman, T. Darrell, The pyramid match kernel: Efficient learning with sets of features, *Journal of Machine Learning Research*, vol. 8, no. 4, pp. 725-760, 2007.

[4] J. Huang, P. C. Yuen, W.-S. Chen, and J. H. Lai, Choosing parameters of kernel subspace LDA for recognition of face images under pose and illumination variations, *IEEE Trans. Syst., Man, Cybern., Part B: Cybern.*, vol. 37, no. 4, pp. 847862, 2007.

[5] H. Xiong, M. N. S. Swamy, and M. O. Ahmad, Optimizing the kernel in the empirical feature space, *IEEE Trans. Neural Netw.*, vol. 16, no. 2, pp. 460474, 2005.

[6] C. Cortes, M. Mohri, A. Rostamizadeh, Learning non-linear combinations of kernels, *Advances in neural information processing systems*, pp. 396404, 2009.

[7] Y. Bengio and Y. Lecun, Scaling Learning Algorithms Toward AI, *Cambridge, MA: MIT Press*, pp. 141, 2007.

[8] G. Hinton and R. Salakhutdinov, Reducing the dimensionality of data with neural networks, *Science*, vol. 313, no. 5786, pp. 504507, 2006.

[9] N. L. Roux and Y. Bengio, Deep belief networks are compact universal approximators, *Neural Comput.*, vol. 22, no. 8, pp. 21922207, 2010.

[10] P. P. Brahma, D. Wu, Y. She, Why Deep Learning Works: A Manifold Disentanglement Perspective, 2015.

[11] F. Anselmi, L. Rosasco, C. Tan, T. Poggio, Deep convolutional networks are hierarchical kernel machines, *arXiv preprint arXiv*, pp. 1508-01084, 2015.

[12] R. Salakhutdinov, G. Hinton, Using deep belief nets to learn covariance kernels for Gaussian processes, *Advances in Neural Information Processing Systems*,no. 20, pp.1249-1256, 2008.

[13] R. Calandra, J. Peters, C. E. Rasmussen, M. P. Deisenroth, Manifold Gaussian processes for regression, *arXiv preprint*, pp. 1402-5876, 2014.

[14] W. Huang, D. Zhao, F. Sun, H. Liu, and E. Chang, Scalable gaussian process regression using deep neural networks, *The 24th International Conference on Articial Intelligence, AAAI Press*, pp. 3576-3582, 2015.

[15] A. G. Wilson, Z. Hu, R. Salakhutdinov, and E. P. Xing, Deep kernel learning, *arXiv preprint*, pp. 1511-02222, 2015.

[16] A. G. Wilson and R. P. Adams, Gaussian process kernels for pattern discovery and extrapolation, *International Conference on Machine Learning (ICML)*, 2013.

[17] A. G. Wilson, H. Nickisch, Kernel interpolation for scalable structured Gaussian processes (KISS-GP), *International Conference on Machine Learning (ICML)*, 2015.

[18] A. G. Wilson, C. Dann, and H. Nickisch, Thoughts on massively scalable Gaussian processes, *arXiv pre-print*, pp. 1511-01870, 2015.

[19] W. K. Wong, M. Sun, Deep Learning Regularized Fisher Mappings, *IEEE Trans. on Neural Networks*, vol. 22, no. 10, pp.1668-75, 2011.

[20] Q. Le, T. Sarlos, and A. Smola, Fastfood-computing Hilbert space expansions in loglinear time, *Proceedings of the 30th International Conference on Machine Learning*, pp. 244-252, 2013.

[21] Z. Yang, M. Moczulski, M. Denil, N. Freitas, A. Smola, L. Song, Z. Wang, Deep fried convnets, *arXiv preprint*, arXiv:1412.7149, 2014.

[22] Y. Cho, L. Saul, Kernel methods for deep learning, *Advances in neural information processing systems*, pp. 342350, 2009.

[23] J. Zhuang, I. Tsang, S. Hoi, Two-layer multiple kernel learning, *International conference on artificial intelligence and statistics*, pp 909917, 2011

[24] E. Strobl, S. Visweswara, Deep multiple kernel learning, *International conference on machine learning and applications*, pp. 414417, 2013.

[25] D. Erhan, Y. Bengio, A. Courville, P. Manzagol, P. Vincent, Why does unsupervised pre-training help deep learning?, *J. Mach. Learn. Res.* no. 11, pp. 625660, 2010.

[26] S.T. Wang, F.L. Chung, J. Wu, J.Wang, Least learning machine and its experimental studies on regression capability, *Appl. Soft Comput.*, no. 21, pp. 677684, 2014.

[27] B. Mitchell, J. Sheppard, Deep structure learning: beyond connectionist approaches, *Proc. 11th International Conference on Machine Learning and Applications*, pp. 162167, 2012.

[28] D. Erhan, A. Courville, and P. Vincent, Why does unsupervised pretraining help deep learning?, *J. Mach. Learn. Res.*, vol. 11, pp. 625660, 2010.

[29] G. Inton, Deep belief networks, *[J]. Scholarpedia*, vol. 4, no. 6, pp. 786-804, 2009.

[30] B. Kwole, (2005) Face detection using convolutional neural networks and Gabor filters, *Artificial neural networks: biological inspirationsICANN, Springer Berlin*, pp. 551556, 2005.

[31] J. Laserson, From neural networks to deep learning: zeroing in on the human brain, *ACM Crossroads Stud Mag*, vol. 18, no. 1, pp. 2934, 2011.

[32] C. Neubauer, Shape position and size invariant visual pattern recognition based on principles of recognition and perception in artificial neural networks, *North Holland, Amsterdam*, pp. 833837, 1992.

[33] P. Vincent, H. Larochelle, Y. Bengio, et al, Extracting and composing robust features with denoising autoencoders, *The 25th ICML2008, ACM Press, New York*, pp. 10961103, 2008

[34] F. J. Huang, Y. L. Cun, Large-scale learning with SVM and convolutional for generic object categorization, *IEEE computer society conference on computer vision and pattern recognition, IEEE Computer Society, Washington*, pp. 284291, 2006.

[35] A. Krizhevsky I. Sutskever, E. Geoffrey, G. E. Hinton, N. Srivastava and R. R. Salakhutdinov, *Improving neural networks by preventing coadaptation of feature detectors*, arXiv preprint arXiv:1207.0580, 2012.

[36] J. Zhuang, I. Tsang, S. Hoi, Two-layer multiple kernel learning, *International conference on artificial intelligence and statistics*, pp. 909917, 2011.

[37] M. Kloft, U. Brefeld, S. Sonnenburg, P. Laskov, K. R.Muller, A. Zien, Efficient and accurate Lp-norm multiple kernel learning, *Adv. Neural Inf. Process Syst.*, vol. 22, no. 22, pp. 9971005, 2009.

[38] A. Rakotomamonjy, E. R. Bach, S. Canu, Y. Grandvalet, Simple MKL, *J. Mach Learn Res*, no. 9, pp. 24912512, 2008.

[39] M. Varma, B. Babu, More generality in efficient multiple kernel learning, *International conference on machine learning*, pp. 10651072, 2009.

[40] F. Bach, G. Lanckriet, M. Jordan M, Multiple kernel learning conic duality and the SMO algorithm, *International conference on machine learning*, pp. 19, 2004.

[41] S. Sonnenburg, G. Rtsch, C. Schfer, B. Schlkopf, Large scale multiple kernel learning, *J. Mach Learn Res* no. 7, pp. 15311565, 2006.

[42] M. Kloft, U. Brefeld, S. Sonnenburg, A. Zien, Lp-norm multiple kernel learning, *J. Mach Learn Res*, no. 12, pp. 953997, 2011.

[43] G. E. Hinton, R. R. Salakhutdinov, Reducing the dimensionality of data with neural networks, *[J]. Science*, vol. 313, no. 5786, pp. 504-507, 2006.

[44] H. Lee, R. Grosse, R. Ranganath, A. Ng, Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations, *The 26th International Conference on Machine Learning*, 2009.

[45] J. Lee, J. H. Lim, H. Choi, et al, Multiple Kernel Learning with Hierarchical Feature Representations, *Neural Information Processing, Springer Berlin Heidelberg*, pp. 517-524, 2013.

[46] F. Orabona,L. Jie, Ultra-fast optimization algorithm for sparse multi kernel learning, *Conference on Machine Learning*, 2011.

[47] A. G. Wilson, Z. Hu, R. Salakhutdinov, et al, Deep Kernel Learning, *[J]. Computer Science*, 2015.

[48] A. G. Wilson, R. P. Adams, Gaussian process kernels for pattern discovery and extrapolation, *International Conference on Machine Learning*, 2013.

[49] C. S. Cruz and J. R. Dorronsoro, A nonlinear discriminant algorithm for feature extraction and data classification, *IEEE Trans. Neural Netw.*, vol. 9, no. 6, pp. 13701376, 1998.

[50] A. Stuhlsatz, J. Lippel, and T. Zielke, Discriminative feature extraction with deep neural networks, *Int. Joint Conf. Neural Netw., Barcelona, Spain*, pp. 1823, 2010.

[51] A. Stuhlsatz, J. Lippel, and T. Zielke, Feature extraction for simple classification, *20th Int. Conf. Pattern Recognit*, Istanbul, Turkey, pp. 15251528, 2010.

[52] W. K. Wong, M. Sun, Deep Learning Regularized Fisher Mappings, *[J]. IEEE Trans. on Neural Networks*, vol. 22, no. 10, pp. 1668-75, 2011.

[53] S. Haykin, Neural Networks: A Comprehensive Foundation, *2nd ed. Upper Saddle River, NJ: Pearson Education*, 1999.

[54] S. Ding, L. Guo, Y. Hou, Extreme learning machine with kernel model based on deep learning, *[J]. Neural Computing and Applications*, pp. 1-10, 2016.

[55] N. Kumar, R. D. Sharma, A. Karmakar, et al, Deep Learning-Based Image Kernel for Inductive Transfer, *[J]. Computer Science*, 2015.

[56] Z. Liu, Kernelized Deep Convolutional Neural Network for Describing Complex Images, *[J]. Feminism and Psychology*, vol.5, no. 4, pp. 481-486, 2015.

[57] S. Poria, E. Cambria, A. Gelbukh, Deep Convolutional Neural Network Textual Features and Multiple Kernel Learning for Utterance-Level Multimodal Sentiment Analysis, *Conference on Empirical Methods in Natural Language Processing*, 2015.

[58] K. Yu, W. Xu, Y. Gong, Deep Learning with Kernel Regularization for Visual Recognition, *[J]. Advances in Neural Information Processing Systems*, pp. 1889-1896, 2008.

[59] J. Zhuang, I. W. Tsang, S. C. H. Hoi, Two-Layer Multiple Kernel Learning, *Journal of Machine Learning Research*, vol. 15, no. 15, pp.909-917, 2011.

[60] I. Rebai, Y. Benayed, W. Mahdi, Deep multilayer multiple kernel learning, *J. Neural Computing and Applications*, pp. 1-10, 2015.

[61] S. Wang, Y. Jiang, F. L. Chung, et al, Feedforward kernel neural networks generalized least learning machine and its deep learning with application to image classification, *[J]. Applied Soft Computing*, vol. 37, pp. 125-141, 2015.

[62] K.L. Du, M.N.S. Swamy, Neural Networks in a Softcomputing Framework, *Springer-Verlag*, 2006.

[63] S.T. Wang, F.L. Chung, J. Wu, J.Wang, Least learning machine and its experimental studies on regression capability, *Appl. Soft Comput.* no.21, pp. 677684, 2014.

[64] G.B. Huang, X. Ding, H. Zhou, Optimization method based extreme learning machine for classification, *Neurocomputing*, vol. 74, no. 13, pp. 155163, 2010.

[65] Y. Cho, L.K. Saul, Kernel Methods for Deep Learning, Advances in Neural Information Processing Systems, *Conference on Neural Information Processing Systems 2009*, Vancouver, British Columbia, Canada, pp. 342-350 2009.