Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 132 (2018) 679–688

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Computational Intelligence and Data Science (ICCIDS 2018)

Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach

Sakshi Indolia[a,*], Anil Kumar Goswami[b], S. P. Mishra[b], Pooja Asopa[a]

[a]*Banasthali Vidyapeeth, Rajasthan, India*
[b]*DRDO, Delhi, India*

Abstract

Deep learning has become an area of interest to the researchers in the past few years. Convolutional Neural Network (CNN) is a deep learning approach that is widely used for solving complex problems. It overcomes the limitations of traditional machine learning approaches. The motivation of this study is to provide the knowledge and understanding about various aspects of CNN. This study provides the conceptual understanding of CNN along with its three most common architectures, and learning algorithms. This study will help researchers to have a broad comprehension of CNN and motivate them to venture in this field. This study will be a resource and quick reference for those who are interested in this field.

Keywords: Convolution Neural Network; Deep Neural Network; Gradient Descent; ADAM

* Sakshi Indolia. Tel.: +91-954-079-5279
E-mail address: sakshiindolia95@gmail.com

1. Introduction

With the rapidly growing demand for learnable machines for solving many complex problems, deep learning has evolved itself as an area of interest to the researchers in the past few years. As researchers tend to mimic human behavior, a major question arises that how do the humans acquire knowledge? The answer to this question is an essential ability of humans i.e. learning, which needs to be incorporated in machines, hence the term machine learning was coined. Machine learning promises to reduce the efforts by making the machines to learn themselves through past experiences [2] using three approaches of learning namely, learning under supervision, without supervision and semi-supervised learning [4]. The conventional machine learning techniques need feature extraction

10.1016/j.procs.2018.05.069

as the prerequisite, and this requires a domain expert [16]. Furthermore, selection of appropriate features for a given problem is a challenging task. Deep learning techniques overcome the problem of feature selection by not requiring pre-selected features but extracting the significant features from raw input automatically for a problem in hand [24]. Deep learning model consists of a collection of processing layers that can learn various features of data through multiple levels of abstraction [15]. Multiple levels allow the network to learn distinct features. Deep learning has emerged as an approach for achieving promising results in various applications like image recognition [31], speech recognition [9], topic classification, sentiment analysis [27], language translation, natural language understanding, signal processing [40], face recognition [13], prediction of bioactivity of small molecules [38] etc. There are different deep learning architectures such as deep belief networks, recurrent neural networks, convolution neural networks etc.

Convolution Neural Network (CNN), often called ConvNet, has deep feed-forward architecture and has astonishing ability to generalize in a better way as compared to networks with fully connected layers [25]. [8] describes CNN as the concept of hierarchical feature detectors in biologically inspired manner. It can learn highly abstract features and can identify objects efficiently [42]. The considerable reasons why CNN is considered above other classical models are as follows. First, the key interest for applying CNN lies in the idea of using concept of weight sharing, due to which the number of parameters that needs training is substantially reduced, resulting in improved generalization [1]. Due to lesser parameters, CNN can be trained smoothly and does not suffer overfitting [32]. Secondly, the classification stage is incorporated with feature extraction stage [16], both uses learning process. Thirdly, it is much difficult to implement large networks using general models of artificial neural network (ANN) than implementing in CNN [36]. CNNs are widely being used in various domains due to their remarkable performance [39] such as image classification ([12]; [41]; [5]), object detection [34], face detection [35], speech recognition [30], vehicle recognition [20], diabetic retinopathy [29], facial expression recognition [37] and many more. The motivation of this study is to establish a theoretical framework while adding to the knowledge and understanding about CNN. The purpose of this study is to present the amalgamation of the elementary principles of CNN and providing the details about the general model, three most common architectures and learning algorithms. A new learning technique, ADAM proposed by [11] has also been elucidated. In addition to that, it computes learning rate for every individual parameter. The complete structure of the sections is as follows. Section 1 gives the introduction and provides the purpose of the study. Section 2 describes the general model of CNN along with all its elementary concepts. Section 3 introduces various architectures of CNN. Section 4 portrays the learning algorithm and Section 5 illustrates the conclusion and future scope.

2. General Model Of Convolution Neural Network

2.1. General Model

The typical model of ANN has single input and output layer along with multiple hidden layers[17]. A particular neuron takes input vector X and produces output Y by performing some function F on it, represented by general equation (1) given by [16] shown below.

$$F(X, W) = Y \tag{1}$$

where, W denotes the weight vector which represents the strength of interconnection between neurons of two adjacent layers. The obtained weight vector can be now used to perform image classification. A significant amount of literature exists related to pixel based classification of images. However, contextual information like shape of the image produces better results or outperforms [21]. CNN is a model that is gaining attention because of its classification capability based on contextual information. The general model of CNN has been described below in figure 1. A general model of CNN consists of four components namely (a) convolution layer, (b) pooling layer, (c) activation function, and (d) fully connected layer. Functionality of each component has been illustrated below.

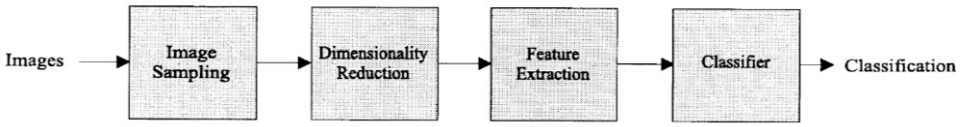

Figure 1: Elementary constituents of CNN [13]

2.2. Convolution Layer

An image to be classified is provided to the input layer and output is the predicted class label computed using extracted features from image [6]. An individual neuron in the next layer is connected to some neurons in the previous layer, this local correlation is termed as receptive field [25]. The local features from the input image are extracted using receptive field.[16]. The receptive field of a neuron associated to particular region in previous layer forms a weight vector, which remains equal at all points on the plane, where plane refers to the neurons in the next layer [13]. As the neurons in plane share same weights, thus the similar features occurring at different locations in the input data can be detected [26]. This has been depicted in figure 2 shown below.

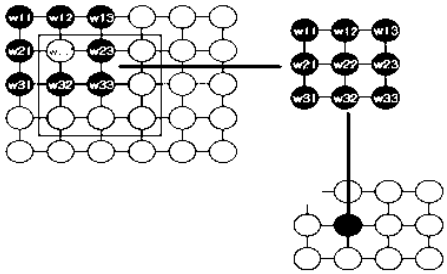

Figure 2: Receptive field of particular neuron in the next layer

The weight vector, also known as filter or kernel, slides over the input vector to generate the feature map [17]. This method of sliding the filter horizontally as well as vertically is called convolution operation. This operation extracts N number of features from the input image in a single layer representing distinct features, leading to N filters and N feature maps [26]. Due to the phenomenon of local receptive field, number of trainable parameters is significantly reduced [43]. The output a_{ij} in the next layer for location (i,j), is computed after applying convolution operation using formula given by [21] as shown below:

$$a_{ij} = \sigma\big((W * X)_{ij} + b\big) \quad (2)$$

where, X is the input provided to the layer, W is filter or kernel which slides over input, b is the bias , * representing the convolution operation, and σ is non-linearity introduced in the network.

2.3. Pooling Layer

The exact location of a feature becomes less significant once it has been detected [16]. Hence, the convolution layer is followed by pooling or sub-sampling layer [13]. The major advantage of using pooling technique is that it remarkably reduces number of trainable parameters and introduces translation invariance [43]. To perform pooling operation, a window is selected and the input elements lying in that window are passed through a pooling function as shown in figure 3.

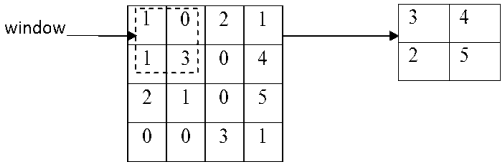

Figure 3: Pooling operation performed by choosing a 2 x 2 window

The pooling function generates another output vector. There exist few pooling techniques like average pooling and max-pooling, out of which max-pooling is the most commonly used technique which reduces map-size very significantly [17]. While computing errors, the error is not back-propagated to winning unit because it does not take part forward flow.

2.4. Fully Connected Layer

Fully connected layer is similar to the fully connected network in the conventional models. The output of the first phase (includes convolution and pooling repetitively) is fed into the fully connected layer, and dot product of weight vector and input vector is computed in order to obtain final output [43]. Gradient descent, also known as batch mode learning or offline algorithm, reduces the cost function by estimating the cost over an entire training dataset, and updates the parameters only after one epoch, where an epoch corresponds to traversing the entire dataset. It yields global minima but if the size of training dataset is large, the time required to train the network substantially increases. This approach of reducing the cost function was replaced by stochastic gradient descent.

2.5. Activation Function

A vast literature exists which uses sigmoid activation function in the conventional machine learning algorithms. In order to introduce non-linearity, use of Rectified Linear Unit (ReLU) has proved itself better than the former, because of two major factors. First, calculation of partial derivative of ReLU is easy [43]. Second, while considering training time to be one of the factor, the saturating non-linearities like sigmoid represented by $f(x) = (1 + e-x-1$are slower than non-saturating non-linearities like ReLU represented by $fx=$ max(0,x) [12]. Third, ReLU does not allow gradients to disappear. But efficiency of ReLU deteriorates when a large gradient is flowing through the network, and update in weight causes the neuron not to get activated leading to Dying ReLU problem which is a considerable issue that is often caused. This issue can be resolved using Leaky ReLU, if x>0, the function activates as f(x)= x and if x<0, the function activates as αx, where α is a small constant.

3. Architectures Of Convolution Neural Network

Various architectures have been developed and implemented in CNN. Brief explanations of those architectures are explained below.

3.1. LeNet Architecture

The multi-layer networks are suitable for image recognition task because of the ability to learn from highly complex and high dimensional data. In 1998, [16] proposed an architecture called as LeNet architecture which uses dataset [14], has been summarized in the following paragraph. The LeNet5 architecture has been depicted in figure 4. It has eight layers constituting five convolutional layers and three fully connected layers. Every unit in a plane has 25 inputs. Units in first hidden layer receives input from 5 × 5 area, which is a part of an entire image thus a very small region of the input image is passed to first hidden layer. This local area of input image is called receptive field

of the unit. Every unit in a plane shares same weight vector. The output of unit is stored at the same location in the feature map.

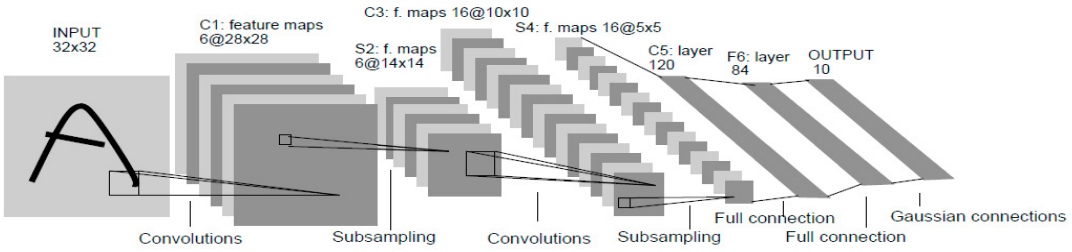

Figure 4 : Architecture of LeNet5, a CNN where each box represents a different feature map.[16]

The neighboring units in the feature map are the result of the neighboring units in the previous layer. Thus, the result is overlapping of contiguous receptive fields. The first layer is convolution layer that consists of neurons which outputs sigmoid activation applied on the weighted sum [36]. As shown in figure 4, while computing contiguous units in the feature map, if 5 × 5 area is chosen as an input and a horizontal shift on the area is performed, it will result to overlapping of four rows and five columns. Various feature maps are generated which are the result of different weight vectors applied to the same input image. Different features can be extracted from the feature maps obtained. Essential property of CNN is that slight shift in the input does not affect feature map.

The precision of the position of the feature in an image is not crucial, thus to reduce the precision value sub-sampling is performed. As shown in figure 4, sub-sampling has been depicted in the second layer. Number of feature maps obtained after sub-sampling are same as those obtained after convolution. Here, in sub-sampling layer 2 × 2 area has been taken as input and compute average of the four inputs, multiply it by trainable coefficient and add the trainable bias, pass it to sigmoid function. Increase in number of feature maps can be observed as the spatial resolution decreases layer by layer. The learning is accomplished with back propagation method.

3.2. AlexNet Architecture

Brief description of a modified variant of LeNet i.e. AlexNet architecture proposed by [12] has been summarized in this fragment. It consists of three fully connected and five convolution layers, the outputs received are passed to the 1000-way softmax in order to classify 1.2 million high resolution images into 1000 distinct classes.

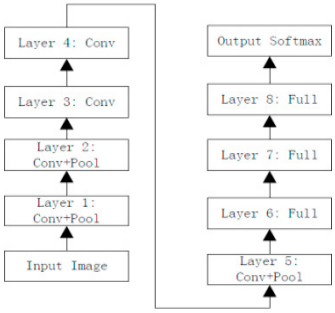

Figure 5: AlexNet architecture [19]

Non-saturating neurons are used along with implementation of efficient GPUs to train the network with faster speed. So, as to make the network classify objects from millions of images, a large network is required which might ultimately lead to an intense demand of training a very large number of weights, leading to the problem of overfitting. To overcome this problem dropout method was implemented. In this technique, the neuron that has

probability 0.5, does not take part in forward and backward propagation, and are damped. The neurons that are dependent on these damped neurons are compelled to learn most robust features completely on their own, which reduces overfitting substantially. The number of iterations required to converge are doubled due to dropout method. The network uses two GTX 580 3GB GPUs and it takes five to six days to train the network. The major features of this architecture were the introduction of ReLU non-linearity in CNN due to which convergence rate increased rapidly [19].

3.3. GoogleNet Architecture

A model commonly known as GoogleNet was proposed by [33]. It was enlightened after winning ILSVRC14 competition. The major aim was to develop a model within lower budget, which could reduce the power, number of trainable parameters used, and memory consumption. The model abruptly reduced the number of trainable parameters used in the network. The broad description of the architecture is as follows. It essentially uses 12 million less parameters than model proposed by [12]. The architecture aimed at building a network that could recognize the objects in an image with more precision. This could possibly be achieved by increasing the size of the network thereby increasing the number of layers, but major drawback of this concept was that if the network size is increased it would increase the number of parameters to be trained thus leading to problem of overfitting.

Another major disadvantage is that if the number of filters is increased, the computation also increases leading to increase in overhead. The solution provided to it was to implement sparse matrix. The units that are highly correlated combine to form a cluster in the previous layer and they provide input to the next layer, hence can be used to achieve optimal network topology [33]. The non-uniform sparse matrix can also be used but the overhead of cache misses still exist even if the computations are reduced up to 100 times. The use of highly tuned numerical libraries performing faster computations also does not help. Therefore, the current state of art works on uniform sparse matrix.

4. Learning Algorithm

Learning Algorithms often called optimizing algorithms benefit the network by minimizing the objective function (often called loss function E(x)), dependent on various learnable parameters like weight, bias etc. Majorly the optimization algorithms can be branched into two categories i.e. first order optimization algorithms and second order optimization algorithms. The First Order Optimization include the computation of gradient represented by Jacobian matrix, the widely used technique is Gradient Descent. There exist a number of variants of Gradient Descent like Mini Batch Gradient Descent and Stochastic Gradient Descent. In order to improve results, improvements have done in the variants such as introduction of momentum, Adagrad, AdaDelta.. Whereas Second Order Optimization include second order derivative represented by Hessian Matrix. One such technique is Adam Optimization. Gradient Descent and Adam Optimization have been briefly explained in the following section.

4.1. Gradient Descent

While training the filters, error backpropagation is the mechanism used to modify the pre-initialized parameters of a network to achieve the optimized network parameters which can produce outputs close to target outputs. In CNN, such network can be achieved using error-backpropagation [16]. As the overall network is a feed forward network, the process starts by computing the outputs at every layer one by one and calculate the error component introduced on the last layer. Now, in order to obtain an optimized network, the computed gradients are backpropagated. Perform the same steps until the efficacy is observed.

Initially input vector is provided to the network. Now, perform convolution operation on the input vector using equation 3 similar and detailed equation is depicted in equation 4. The diagrammatic representation of convolution operation is depicted in figure 5.

$$C_q^l = \left(\sum_{p=1}^{n} S_p^{l-1} * k_{p,q}^l + b_q^l\right) \tag{3}$$

$$C_q^l = \left(\sum_{p=1}^{n} \sum_{u=-x}^{x} \sum_{v=-x}^{x} S_p^{l-1}(i-u, j-v) \cdot k_{p,q}^l(u,v) + b_q^l\right) \quad (4)$$

where, n represents number of feature maps in last layer, p and q denotes feature map indices of current layer and previous layer respectively, ϕ denotes the activation function applied for example, ReLU and sigmoid, l denotes the layer, * denotes convolution operation, b and x represents the bias and size of filter respectively. Initially S_p^0 represents the input image on which first convolution is to be performed and S_p^1 represents the input on which second convolution is to be performed, which can be obtained after applying pooling on S_p^0.

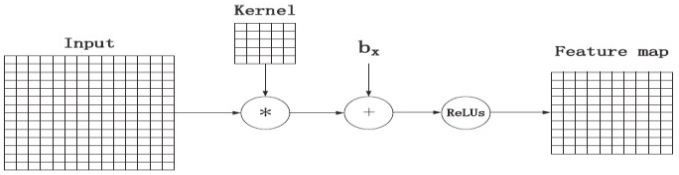

Figure 6 : Convolution operation on 14 x 14 input image by sliding 5 x 5 kernel which yields 10 x 10 feature map. [7]

After convolution operation, pooling is applied in order to introduce translation invariance using equation 5.

$$S_q^l(i,j) = \frac{1}{4} \sum_{u=0}^{z} \sum_{v=0}^{z} C_q^l(2i-u, 2j-v) \quad (5)$$

where, z represents the pool window size. Perform significant number of iterations for convolution and pooling according to the requirement. Pass the result obtained in the last layer of the pooling through fully connected layer for classification and compute the output produced using equation 6.

$$\widehat{output} = \sigma(W \times f + b) \quad (6)$$

where, f is the final output vector obtained after last pooling operation, W is the weight vector of fully connected layer. A classifier can be used on the last layer. A well known classifier softmax, depicted by equation 7 can be used. Where, label represents number of class labels

$$\hat{y}(i) = \frac{e^{\widehat{output}}}{\sum_1^{labels} e^{\widehat{output}}} \quad (7)$$

Compute the loss function using equation 8 which shows the error component introduced in the network.

$$L = \frac{1}{2} \sum_{i=1}^{\substack{noof \\ training \\ patters}} \left(\hat{y}(i) - y(i)\right)^2 \quad (8)$$

where $\hat{y}(i)$ is the target output and $y(i)$ is the obtained output. Now, the error component needs to be backpropagated so that every neuron gets the penalty. The first order derivative wrt weight and bias are computed using equation

$$\Delta W(i,j) = \frac{\partial L}{\partial W(i,j)} \quad (9)$$

$$= \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial W(i,j)} \quad (10)$$

$$= \frac{\partial \left(\frac{1}{2} \sum_{i=1}^{p^*} \left(\hat{y}(i) - y(i)\right)^2\right)}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial W(i,j)} \quad (11)$$

where p^* denotes no of training pattern.

$$= \left(\hat{y}(i) - y(i)\right) \times \frac{\partial}{\partial W(i,j)} \left(\sigma\left(\sum_{j=1}^{\substack{noof \\ nodes}} W(i,j) \times f(j) + b(i)\right)\right) \quad (12)$$

For sigmoid activation function equation 10 will be the result.

$$\Delta \hat{y}(i) = \left(\hat{y}(i) - y(i)\right) \cdot \hat{y}(i)\left(1 - \hat{y}(i)\right) \quad (13)$$

Calculation of ΔW,

$$\Delta W(i,j) = \Delta\hat{y}(i) \times f(j) \quad (14)$$

Calculation of Δb,

$$\Delta b = \frac{\partial L}{\partial b(i)} \quad (15)$$

The summarized algorithm for minimizing objective function using error-backpropagation has been shown below ([43]; [7]).

Step 1: Provide input vector to the network.
Step 2: Perform convolution using filter to produce a feature map.
Step 3: Pass the obtained feature map through ReLU to introduce non-linearity.
Step 4: Apply pooling operation on obtained feature map, which introduces translation invariance.
Step 5: Repeat Steps 2 to 4 for repetition of layers.
Step 6: The obtained feature maps are passed to fully connected layer for classification.
Step 7: Pass the output to a classifier such as softmax.
Step 8: Computer loss at the final layer and calculate gradient w.r.t. all the learnable parameters.
Step 9: Backpropagate the error component and update the parameters.
Step10: Perform the forward pass and repeat Steps 2 to 9 using updated parameters until network converges.

Figure 7 : Summarized algorithm to minimize the cost function.

4.2. Adaptive Moment Estimation (ADAM) Optimization

Gradient Descent is applicable in the scenarios where the function is easily differentiable with respect to the parameters used in the network. According to [16], it is easy to minimize continuous function than minimizing discrete functions. The weight update is performed after one epoch, where one epoch represents running through an entire dataset. This technique produces satisfactory results but it deteriorates if the training dataset size becomes large and does not converge well [3]. It also may not lead to global minimum in case of existence of multiple local minima. Stochastic gradient descent overcomes this drawback by randomly selecting data samples and updating the parameters based on the cost function. Additionally, it converges faster than regular gradient descent and saves memory by not accumulating the intermediate weights [22].

Adaptive Moment Estimation (ADAM), proposed by [11], facilitates computation of learning rates for each parameter using first and second moment of gradient. The summarized description of the same has been explained. Being computationally efficient, ADAM requires less memory and outperforms on large datasets. It require $\boldsymbol{p_o}$, $\boldsymbol{q_o}$, t to be initialized to 0, where $\boldsymbol{p_o}$ corresponds to $\mathbf{1^{st}}$ moment vector i.e. mean, $\boldsymbol{q_o}$ corresponds to $\mathbf{2^{nd}}$ moment vector i.e. uncentered variance and t represents timestep. While considering $\boldsymbol{f(w)}$ to be the stochastic objective function with parameters $\boldsymbol{w}$, proposed values of parameters in ADAM according to [11], are as follows: $\boldsymbol{\alpha = 0.001}$,$\boldsymbol{m_1 = 0.9}$, $\boldsymbol{m_2 = 0.999}$, $\boldsymbol{\in = 10^{-8}}$. Another major advantage discussed in the study of ADAM is that the updation of parameter is completely invariant to gradient rescaling, the algorithm will converge even if objective function changes with time. The drawback of this particular technique is that it requires computation of second order derivative which results in increased cost. The algorithm of ADAM has been briefly mentioned below.

Step 1: while w_t do not converges

do{

Step 2: Calculate gradient $g_t = \frac{\partial f(x,w)}{\partial w}$ (16)

Step 3: Calculate $p_t = m_1 \cdot p_{t-1} + (1 - m_1) \cdot g_t$ (17)

Step 4: Calculate $q_t = m_2 \cdot q_{t-1} + (1 - m_2) \cdot g_t^2$ (18)

Step 5: Calculate $\hat{p}_t = p_t/(1 - m_1^t)$ (19)

Step 6: Calculate $\hat{q}_t = q_t/(1 - m_2^t)$ (20)

Step 7: Update the parameter $w_t = w_{t-1} - \alpha \cdot \hat{p}_t/\left(\sqrt{\hat{q}_t} + \in\right)$ (21)

}

Step 8: return w_t

5. Conclusion

Major advantage of deep learning over conventional machine learning technique is that it can independently detect relevant features in high dimensional data as compared to shallow networks. There exists sufficient literature on different deep learning techniques such as recurrent neural network, deep belief networks and CNN. This study has thrown light upon the basic understanding of CNN, which is a deep learning approach to solve many complex problems. This study has described the general model, various architectures, and two important learning algorithms of the CNN. CNN has emerged as a prominent technique used for classification based on contextual information. It has immense ability to learn contextual features and thereby has overcome the issues involved in pixel wise classification. It reduces number of parameters required to a great extent. CNN is extensively being used for classification in remote sensing [21], ocean front recognition task [18], high-resolution data, traffic sign recognition [10], audio scene [28], segmentation of MR brain images [23]. This study will provide broad understanding to researchers who want to venture in this field. It will act as a means to learners, researchers and to those who are interested in this field.

References

[1] Arel, I., Rose, D. C., and Karnowski, T. P. (2010) "Deep machine learning-a new frontier in artificial intelligence research [research frontier]." *IEEE computational intelligence magazine* **5** (**4**): 13-18.

[2] Carbonell, J. G., Michalski, R. S., and Mitchell, T. M. (1983) "An overview of machine learning. In Machine learning." *Springer Berlin Heidelberg* (pp. 3-23).

[3] Chen, H., Tang, Y., Li, L., Yuan, Y., Li, X., & Tang, Y. (2013) "Error analysis of stochastic gradient descent ranking." *IEEE transactions on cybernetics* **43** (**3**): 898-909.

[4] Dharmadhikari, S. C., Ingle, M., and Kulkarni, P. (2011) "Empirical studies on machine learning based text classification algorithms." *Advanced Computing* **2** (**6**): 161.

[5] Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., and Darrell, T. (2014) "Decaf: A deep convolutional activation feature for generic visual recognition." *In International conference on machine learning* (pp. 647-655).

[6] Fang, J., Zhou, Y., Yu, Y., and Du, S. (2017) "Fine-grained vehicle model recognition using a coarse-to-fine convolutional neural network architecture." *IEEE Transactions on Intelligent Transportation Systems* **18** (**7**): 1782-1792.

[7] Feng, J., Li, F., Lu, S., Liu, J., and Ma, D. (2017) "Injurious or Noninjurious Defect Identification From MFL Images in Pipeline Inspection Using Convolutional Neural Network." *IEEE Transactions on Instrumentation and Measurement* **66** (**7**): 1883-1892.

[8] Fieres, J., Schemmel, J., and Meier, K. (2006) "Training convolutional networks of threshold neurons suited for low-power hardware implementation." In *Neural Networks, 2006. IJCNN'06. International Joint Conference on. IEEE*. (pp. 21-28).

[9] Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. (2012) "Improving neural networks by preventing co-adaptation of feature detectors". *arXiv preprint arXiv*:1207.0580.

[10] Jin, J., Fu, K., and Zhang, C. (2014) "Traffic sign recognition with hinge loss trained convolutional neural networks." *IEEE Transactions on Intelligent Transportation Systems* **15** (**5**): 1991-2000.

[11] Kingma, D. P., & Ba, J. (2014). "Adam: A method for stochastic optimization. " *arXiv preprint arXiv:1412.6980*.

[12] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). "Imagenet classification with deep convolutional neural networks." In *Advances in neural information processing systems* (pp. 1097-1105).

[13] Lawrence, S., Giles, C. L., Tsoi, A. C., and Back, A. D. (1997) "Face recognition: A convolutional neural-network approach." *IEEE transactions on neural networks* **8** (**1**): 98-113.

[14] LeCun, Y. (1998). "The MNIST database of handwritten digits." *http://yann. lecun.com/exdb/mnist/*.

[15] LeCun, Y., Bengio, Y., and Hinton, G. (2015) "Deep learning." *Nature* **521** (**7553**): 436-444.

[16] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998) "Gradient-based learning applied to document recognition." *Proceedings of the IEEE* **86** (**11**): 2278-2324.

[17] Lee, K. B., Cheon, S., & Kim, C. O. (2017) "A convolutional neural network for fault classification and diagnosis in semiconductor manufacturing processes." *IEEE Transactions on Semiconductor Manufacturing* **30** (**2**): 135-142.Lima, E., Sun, X.,

[18] Dong, J., Wang, H., Yang, Y., and Liu, L. (2017) "Learning and Transferring Convolutional Neural Network Knowledge to Ocean Front Recognition." *IEEE Geoscience and Remote Sensing Letters* **14** (**3**): 354-358.

[19] Liu, H., Li, B., Lv, X., and Huang, Y. (2017) "Image Retrieval Using Fused Deep Convolutional Features." *Procedia Computer Science* **107**: 749-754.

[20] Luo, X., Shen, R., Hu, J., Deng, J., Hu, L., and Guan, Q. (2017) "A Deep Convolution Neural Network Model for Vehicle Recognition and Face Recognition." *Procedia Computer Science* **107**: 715-720.

[21] Maggiori, E., Tarabalka, Y., Charpiat, G., and Alliez, P. (2017) "Convolutional neural networks for large-scale remote-sensing image classification." *IEEE Transactions on Geoscience and Remote Sensing* **55** (**2**): 645-657.

[22] Manem, H., Rajendran, J., & Rose, G. S. (2012) "Stochastic gradient descent inspired training technique for a CMOS/nano memristive trainable threshold gate array." *IEEE Transactions on Circuits and Systems I: Regular Papers* **59** (**5**): 1051-1060.

[23] Moeskops, P., Viergever, M. A., Mendrik, A. M., de Vries, L. S., Benders, M. J., and Išgum, I. (2016) "Automatic segmentation of MR brain images with a convolutional neural network." *IEEE transactions on medical imaging* **35** (**5**): 1252-1261.

[24] Mrazova, I., and Kukacka, M. (2012) "Can deep neural networks discover meaningful pattern features?." *Procedia Computer Science* **12:** 194-199.

[25] Nebauer, C. (1998) "Evaluation of convolutional neural networks for visual recognition." *IEEE Transactions on Neural Networks* **9** (**4**): 685-696.

[26] Palsson, F., Sveinsson, J. R., and Ulfarsson, M. O. (2017) "Multispectral and Hyperspectral Image Fusion Using a 3-D-Convolutional Neural Network." *IEEE Geoscience and Remote Sensing Letters* **14** (**5**): 639-643.

[27] Pang, B., and Lee, L. (2008) "Opinion mining and sentiment analysis." *Foundations and Trends® in Information Retrieval* **2**(**1**–**2**): 1-135.

[28] Phan, H., Hertel, L., Maass, M., Koch, P., Mazur, R., and Mertins, A. (2017) "Improved Audio Scene Classification Based on Label-Tree Embeddings and Convolutional Neural Networks." *IEEE/ACM Transactions on Audio, Speech, and Language Processing* **25** (**6**): 1278-1290.

[29] Pratt, H., Coenen, F., Broadbent, D. M., Harding, S. P., and Zheng, Y. (2016) "Convolutional neural networks for diabetic retinopathy." *Procedia Computer Science* **90**: 200-205.

[30] Sainath, T. N., Kingsbury, B., Mohamed, A. R., Dahl, G. E., Saon, G., Soltau, H., Ramabhadran, B. (2013) "Improvements to deep convolutional neural networks for LVCSR." *In Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on* (pp. 315-320).

[31] Simonyan, K., and Zisserman, A. (2014) "Very deep convolutional networks for large-scale image recognition". *arXiv preprint arXiv*:1409.1556.

[32] Smirnov, E. A., Timoshenko, D. M., and Andrianov, S. N. (2014) "Comparison of regularization methods for imagenet classification with deep convolutional neural networks."*Aasri Procedia* **6**: 89-94.

[33] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... and Rabinovich, A. (2015) "Going deeper with convolutions." *In Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1-9).

[34] Szegedy, C., Toshev, A., and Erhan, D. (2013) "Deep neural networks for object detection." *In Advances in neural information processing systems* (pp. 2553-2561).

[35] Timoshenko, D., and Grishkin, V. (2013) "Composite face detection method for automatic moderation of user avatars." *Computer Science and Information Technologies* (CSIT'13).

[36] Tivive, F. H. C., and Bouzerdoum, A. (2005) "Efficient training algorithms for a class of shunting inhibitory convolutional neural networks." *IEEE Transactions on Neural Networks* **16** (**3**): 541-556.

[37] Uçar, A. (2017, July) "Deep Convolutional Neural Networks for facial expression recognition." *In Innovations in Intelligent Systems and Applications (INISTA), 2017 IEEE International Conference on* (pp. 371-375).

[38] Wallach, I., Dzamba, M., and Heifets, A. (2015) "AtomNet: a deep convolutional neural network for bioactivity prediction in structure-based drug discovery." *arXiv preprint arXiv*:1510.02855.

[39] Wang, J., Lin, J., and Wang, Z. (2016) "Efficient convolution architectures for convolutional neural network." *In Wireless Communications and Signal Processing (WCSP), 2016 8th International Conference on* (pp. 1-5).

[40] Yu, D., and Deng, L. (2011) "Deep learning and its applications to signal and information processing [exploratory dsp]". *IEEE Signal Processing Magazine*, 28(1), 145-154.

[41] Zeiler, M. D., and Fergus, R. (2013) "Stochastic pooling for regularization of deep convolutional neural networks." *arXiv preprint arXiv*:1301.3557.

[42] Zhang, Z. (2016) "Derivation of Backpropagation in Convolutional Neural Network(CNN)".

[43] Zhou, Y., Wang, H., Xu, F., and Jin, Y. Q. (2016) "Polarimetric SAR image classification using deep convolutional neural networks." *IEEE Geoscience and Remote Sensing Letters* **13** (**12**): 1935-19.