Rethinking Motivation of Deep Neural Architectures

Weilin Luo, Jinhu Lü, *Fellow, IEEE*,
Xuerong Li, Lei Chen, and Kexin Liu

Abstract

Nowadays, deep neural architectures have acquired great achievements in many domains, such as image processing and natural language processing. In this paper, we hope to provide new perspectives for the future exploration of novel artificial neural architectures via reviewing the proposal and development of existing architectures. We first roughly divide the influence domain of intrinsic motivations on some common deep neural architectures into three categories: information processing, information transmission and learning strategy. Furthermore, to illustrate how deep neural architectures are motivated and developed, motivation and architecture details of three deep neural networks, namely convolutional neural network (CNN), recurrent neural network (RNN) and generative adversarial network (GAN), are introduced respectively. Moreover, the evolution of these neural architectures are also elaborated in this paper. At last, this review is concluded and several promising research topics about deep neural architectures in the future are discussed.

Digital Object Identifier 10.1109/MCAS.2020.3027222
Date of current version: 12 November 2020

I. Introduction and Background

Since the concept of artificial intelligence (AI) was proposed in the Dartmouth workshop, AI techniques have given rise to great reforms in various aspects of modern society, such as public opinion analysis in social media, recommender system of e-commerce and intelligent manufacturing in industrial production. In the whole AI community, artificial neural network (ANN) is one class of the most critical algorithms with profound influence. Unlike general networks [1], [2], ANN consists of many connected artificial neurons that model the

function of biological neurons. These artificial neurons transmit nonlinear activation through some weighted connections. Then, the network can approximate a specific complicated function via stacking enough such transmission. Despite the significant achievements of ANNs, they are limited to their capacity of tackling data in the raw form [3]. Sophisticated feature engineering requiring considerable expertise is necessary for the network to learn reliable patterns from the raw data. Inspired by the investigations to hierarchical structures of human speech system, some novel network architectures with multiple layers are developed, namely deep neural networks (DNNs). These hierarchical neural architectures are expected to overcome the limitation of conventional ANNs.

A historical timeline is depicted to help overview the appearance of various influential DNN architectures. As shown in Fig. 1, group method of data handling (GMDH) based network was proposed in 1965 [4]. To the best of our knowledge, it was the first time that multilayer representation learning is introduced in ANNs. Following this sort of hierarchical structure and being inspired by the characteristics found in the visual cortex of cats [5], Fukushima [6] presented a genuine deep neural architecture in 1979, namely neocognitron. This model contains a convolutional structure which has a receptive field with given weights, can capture features from a 2-dimensional input plane. Although the architecture is similar to convolutional neural networks (CNNs) [7], the weights of neocognitron are not trained by backpropagation-based supervised learning. Instead, the weights are given through a winner-take-all-based unsupervised learning strategy [8]. After that, in 1988, autoencoder [9] was developed which can learn a low-dimensional representation of the input data. This network firstly used a self-associated pre-training techniques based on unsupervised learning with an encoder-decoder framework [10], which is one of the widely applied network frameworks today. In 1990, recurrent neural network (RNN) [11] containing the structure of recurrent connection was designed to model the short-term memory for neural networks. This innovation enables RNN to process temporal sequential data effectively. Subsequently, in 1998, convolutional neural network (CNN) [7] was proposed and it achieved impressive performance in the task of handwriting character recognition. With the structure of neocognitron, CNN adopts backpropagation based on gradient descent algorithm to optimize the network weights end to end.

Despite the appearance of early DNNs, the concept of deep learning was not popular until 2006 when Hinton et al. [12] proposed deep belief network (DBN). They exploited an unsupervised training algorithm to optimize DBN layer by layer. This technique allows neural architectures with more layers to be trained efficiently and effectively, thus attracts lots of attention to deep learning. In 2014, the framework of generative adversarial network (GAN) [13] was presented being inspired by the game theory. As a generative model with a dualistic framework [14], the two sub-networks in GAN can be trained through the game between the two networks and can learn data distribution implicitly. After that in 2017, a novel capsule structure was proposed in capsule network (CN) [15]. This structure is a special artificial neuron in which vectors rather than scalars are taken as the information carrier. Such a characteristic enables CN to capture the spatial relation between image patterns, which can hardly be implemented by CNNs.

These deep neural architectures, nowadays, have turned out to be very effective in many scenarios such as video security [16], electronic commerce [17] and intelligent manufacturing [18]. Besides, DNNs also show great potential for the research on complex network [19], bioinformatics [20] and internet of things [21] etc.

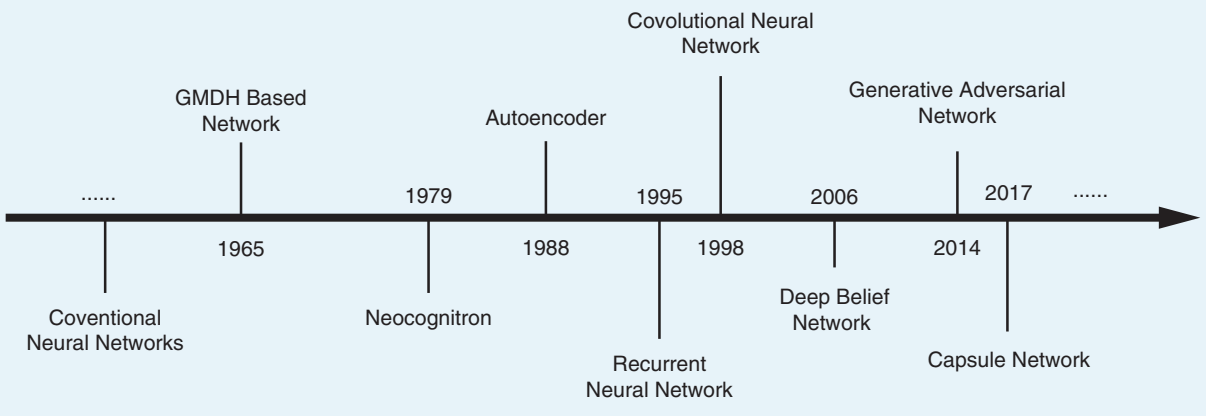


Figure 1. The timeline of DNNs.

(see Fig. 2). Nevertheless, the progress of the AI research seems to be slowing and many advances in neural network algorithms are actually not effective as they are expected [22]. Therefore, it might be a good choice to look back now, reviewing the important advances in DNNs and rethinking how those classic neural architectures are developed and why they can work. We believe that such a research is helpful for further promoting the progress of DNNs. In this paper, we attempt to provide an insight of DNNs from the perspective of their intrinsic motivations. The rest of this paper is organized as follows. In Section II, the influence domain of network motivations is defined and coarsely categorized to help understand the effect of different motivations. Then in Section III-V, convolutional neural networks, recurrent neural networks and generative adversarial networks are introduced respectively to illustrate how the architectures are motivated and constructed. Evolution of the three deep neural architectures are also discussed in these sections. In Section VI, several conclusions about intrinsic motivations of DNNs are summarized and some promising research topics are discussed as well.

II. Influence Domain of Motivation on Deep Neural Architectures

Since the first mathematical model of biological neuron was proposed, new neural networks have been appearing constantly such that listing all network architectures seems to be practically impossible. Considering the diversity of neural architectures' intrinsic motivation, we attempt to find and analyze the general characteristics of motivations in DNNs. For clarity, we introduce the concept of influence domain to represent which part of a neural architecture is inspired and motivated. Through reviewing the proposal and evolution of some deep neural architectures, we roughly divide the influence domain of different network motivations into three categories: information processing, information transmission and learning strategy.

A. Information Processing

Artificial neuron is the basic information processing unit in neural networks. How neurons work and how information is processed in the network directly affect the network's ability of representation learning. Hence, many deep neural architectures are constructed with their own way of information processing. Motivated by the study of visual cortex of cat, a convolutional structure is presented in neocognitron and convolutional neural network (CNN). The convolutional structure consists of neurons arranged in columns, which is termed as convolutional kernel. Such an aggregation of neurons enables neural network to learn representations directly from the raw image, avoiding plenty of feature engineering in conventional machine learning. On this basis, some improved convolutional structures are proposed to enhance the network performance. Dilated convolution [23] allows the receptive field of convolutional kernel to be enlarged, thus can capture more information from larger spatial region. Deformable convolution [24] can learn an offset for each sampling point of convolutional kernel, so the sampling points can be arranged irregularly. This structure leads to stronger robustness to the scale and shape transformation of objects in image.

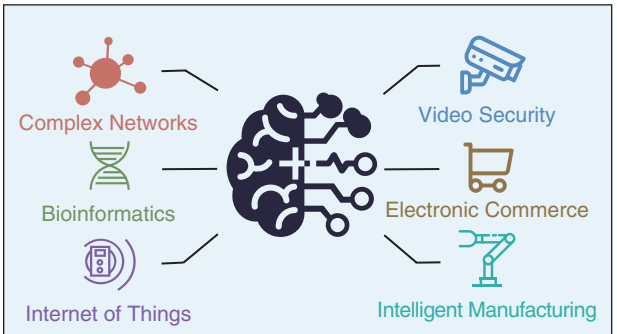


Figure 2. Present and potential application domains of deep neural architectures.

Despite the power of CNN and its variants in feature extraction, they can only capture the existence of patterns. In fact, it is hard for the convolution structure to learn the interrelationship between different patterns [15]. In recent years, it is argued that there are a large number of micro-columns neuron modules in cerebral cortex that can handle different types of visual stimulus. Motivated by this, a capsule structure is invented to solve the problem of CNNs [15]. In contrast to the convolutional kernel that outputs a scalar to measure the existence of a pattern in current location, the capsule produces a vector to represent the attributes of a pattern (such as the location, posture and so on). Therefore, capsule network is more promising in tasks that have higher requirements on feature representation, such as image segmentation and object detection. Moreover, due to the abundant information in the output vector of capsule, neural network is endowed with better interpretability.

B. Information Transmission

Another important aspect of neural architectures is how information flows in the network. Generally, the information transmission path can decide how hidden features are used by neurons in different layers. In comparison of standard CNN, GoogLeNet [25] contains multiple parallel paths between two layers. In each path, features from the previous layer are filtered by convolutional kernels with a specific scale. After that, the outputs of the

convolutional kernels from different paths are concatenated together as the input of the next layer. Consequently, features with different scales can be captured through this structure, which improves the network's capability of object detection. In addition, ResNet [26] is proposed that adds shortcut connections from shallower layer to deeper layer. This shortcut connection is able to ease the training of network with very deep architecture, because it actually equals to an identity mapping which allows the gradient to be propagated to shallower layers without vanishing during the backpropagation.

The information can not only flow forward during the network inference, but also be transmitted in more diverse ways in some neural architectures. These special ways of information transmission lead to an ability of handling data with different forms. To address temporal sequential data, RNN takes as input the data at each time step respectively. The recurrent connection makes hidden state of the previous time step flow to the input of current time step. Thus, RNN can learn a representation from the temporal sequential data, aggregating both the information of the current and that of the past. Furthermore, the recurrent connection can be improved by adding some "gates" to enhance, suppress or even discard the information flow from the previous time step [27]. Another instance is Graph neural network (GNN). This network architecture is able to deal with graph data. Through the edges in the input graph, the information of each node will be aggregated with that of adjacent nodes. Being fed this combination of node information, the neural network can learn a graph embedding in accordance with the topology of graph [28] and realize the classification of each node or the whole graph. With larger field of information aggregation, the information can even flow from farther away nodes to the central node.

C. Learning Strategy

Besides the processing and transmission of information, how a deep neural architecture learns from data is also of great significance. The learning strategy mainly determines what the network can learn and what kind of tasks it can fulfill. For most deep neural architectures, the network parameters can be trained end-to-end through supervised learning technique. It allows the network to learn a mapping from the input data to a desired output. In other words, the network learns a conditional probability density function $p(y|x)$, where x and y denote the data and desired output respectively. On this basis, the strong fitting ability of deep neural architectures makes them popular for many discriminative tasks such as recognition, detection and classification.

Moreover, deep neural architectures can directly learn the distribution of data itself, i.e., $p(x)$. Autoencoder [9] is designed as an encoder-decoder framework which can be used to compress data. The encoder part transforms the data to a low-dimensional representation while the decoder restores the original input from the representation. For this purpose, the entire architecture is trained through a self-supervised learning strategy: minimizing the reconstruction error between the restored data and the raw data. Another famous leaning strategy in recent years is adversarial learning. It is adopted to train the architecture of GAN [13] which contains two networks: generator and discriminator. The objective of adversarial learning is to enable the generator to learn the data distribution implicity through a zero-sum game between the two networks.

According to the three influence domains of intrinsic motivations, we can further discuss how DNNs are motivated and developed. In the following three sections, CNN, RNN and GAN are taken as instances respectively to illustrate the influence of their own motivations to the architecture. Specifically in each section, the intrinsic motivation of a DNN and the influence domain of motivation are discussed firstly in each section. Then, details of the neural architecture are also demonstrated to deepen the understanding of the network structure. At the end of each of next three sections, some variants of the DNN are introduced to provide more knowledge about corresponding neural architectures.

III. From Animal Visual Cortex to CNNs

This section mainly focuses on convolutional neural network (CNN) and how it is motivated from the perspective of information processing.

A. Motivation on Information Processing of CNN

CNN is one of the deep discriminative models, which has been proven to be pretty effective in many computer vision tasks such as object detection [29], object tracking [30] and video classification [31]. The birth of CNN integrates computer science, mathematics and most importantly, neurobiology. In an interesting experiment, Hubel and Wiesel [5] found that there are a small number of cells in the visual cortex that are sensitive to specific vision areas. They discovered that some individual neurons in animals visual system can transmit electrical signal in response to certain properties of visual sensory inputs, such as the edges with a specific orientation [32].

In traditional neural networks, the hidden state is represented as a one-dimensional vector and each neuron takes as input the entire state from the previous layer. For image data, embedding a raw image into an one-dimensional representation actually destroys the spatial characteristics of the data. Inspired by Hubel's study, Lecun et al. [7] designed a neural architecture based on

convolution structure, i.e., convolutional neural network (CNN). The convolution structure embeds the raw image into a group of two-dimensional maps. And, each neuron of CNN directly samples a local area of the map generated by the previous layer. This way of information processing allows the neuron to effectively capture the local spatial characteristics from the raw image. Fig. 3 shows the structure of standard CNN, which is called LeNet-5.

B. Details of Architecture

The whole architecture of LeNet-5 consists of two parts: convolution module and full connection module. The convolution module is composed of 2 convolutional layers, each of them is followed by a sub-sampling layer. In each convolutional layer, there are lots of convolutional units (namely convolutional filters or kernels) shifting at a fixed stride across the input plane (such as a raw image or a 2-D feature maps). As a simulation of those special individual neurons in the animals' visual system, each convolutional kernel gives a strong response when meeting a specific image pattern during the shifting and gives weaker responses to other areas. All these responses form a feature map which reflects the spatial distribution of the corresponding image pattern in the input plane. In other words, the image pattern is transformed by the convolutional kernel from the original form to a representation in a higher level. With the stack of the kernels, more image patterns are transformed to high-level representations and the perception of the raw image can be obtained via combining these representations. Specifically, the convolution module mainly contains three key ideas: local receptive field, weight-sharing and spatial sub-sampling.

- *Local receptive field*. Each convolutional kernel consists of a weight matrix which has a dot product with the input elements located in a small neighborhood (see the black squares in Fig. 3). This structure is similar to a local receptive field of the animal visual cortex organization. Thus, each convolutional kernel focuses on one local pattern of the input plane.
- *Weight sharing*. During the slide of a convolutional kernel across the input plane, the weights of the kernel are shared for each local receptive field of the input. Obviously, much fewer parameters are required during the forward propagation due to this operation in contrast to full-connected neurons.
- *Sub-sampling*. The sub-sampling is to replace elements in a local area of the feature map with their mean value (mean-pooling) or the maximum (max-pooling). Via such an operation, the pattern of the local area is represented by just one element, rather than the spatial arrangement of all elements. Moreover, due to that the number of parameters is also reduced, the risk of overfitting can be lowered while training the network.

The last layer of the convolution module transforms the feature maps of the previous layer to a feature vector. Then, the full connection module, which equals a multilayer perceptron, transforms the high-level feature vector to the discriminated result. Consequently, a mapping from the raw image to the output is constructed via these two modules. Formally, the whole architecture can be represented by a function as follows:

$$\mathbf{y} = F(\mathbf{x}, \{\theta_i\}), \tag{1}$$

where $\mathbf{x}$ and $\mathbf{y}$ are respectively the input and output of CNN. $\{\theta_i\}$ denotes the set of parameters of different layers, which can be learned directly from data via the error-back propagation algorithm or other advanced learning algorithms (such as Levenberg-Marquardt and neuron by neuron) [33].

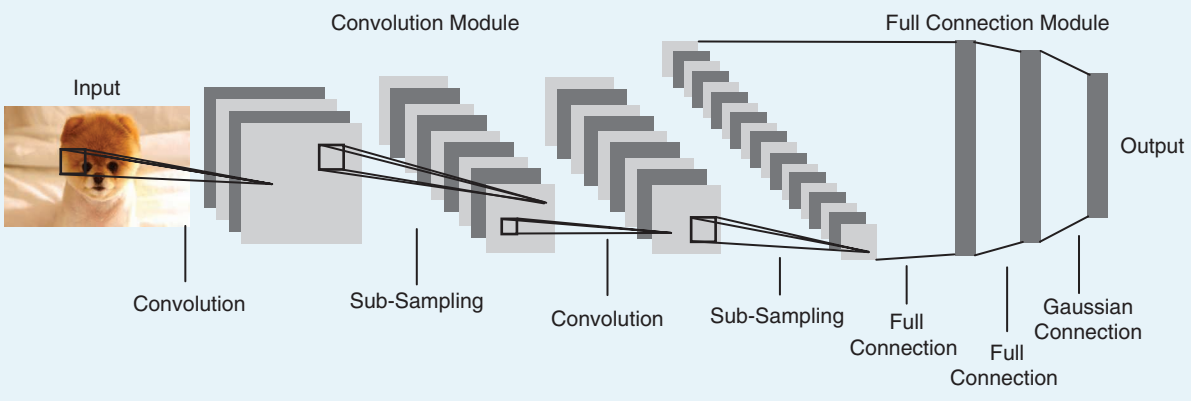


Figure 3. The structure of LeNet-5 consists of two parts: the convolution module and the full connection module [7].

C. Evolution of CNNs

In addition to being motivated by Hubel's discovery [5], CNN can also be improved based on another observation in neuroscience. There is an activation function, commonly *sigmoid* function, performing the nonlinear transformation to the outcome of each convolutional kernel. However, nearly half neurons of the network are activated while using sigmoid function to compute the activation, which goes against the neuroscience observation that only 1% ~ 4% neurons in brain fire simultaneously [34]. Therefore, rectified linear unit (ReLU) is presented as a more accurate activation model of brain neurons that contains three characteristics: one-sided inhibition, wide exciting boundaries and sparsity [35].

In recent years, many variants of CNN have been proposed to improve the network performance. As mentioned in Section II-A and II-B, the receptive field of convolutional kernels are transformed in [23] and [24], which increases the robustness of CNN to the scale and shape transformation of objects. Parallel convolution in GoogLeNet [25] allows the network to capture features with multiple scales and shortcut connection in ResNet [26] makes it possible to train a network with more than 100 layers. Based on ResNet, DenseNet [36] contains denser shortcut connections. Each layer in DenseNet takes as input the feature maps of all the layers in front of it. That is to say, the features of a shallower layer are reused by all the layers behind, which improves the performance further.

In terms of lightweight model, some studies divide the convolution operation into many groups to accelerate the inference through reducing the amount of computation [37], [38]. Given an input feature with size of $64 \times 64 \times 28$ and a convolutional kernel with size of $9 \times 9 \times 16$, then the amount of computation in a convolution is $28 \times 9 \times 9 \times 16 = 36288$. While with the group convolution, the input feature is divided into 2 groups firstly. Then, each group $(64 \times 64 \times 14)$ is filtered by half of the original convolutional kernel $(9 \times 9 \times 8)$ respectively. Therefore, the amount of computation is $14 \times 9 \times 9 \times 8 \times 2 = 18144$. In addition, the parameters can be decreased by the separation of the convolution [39]. This variant separates standard convolution (3-D) to convolution inside each channel (2-D) and 1×1 convolution among channels (1-D).

Besides, attention mechanism is also adopted to improve CNN's ability of capturing the key information. In general, the attention mechanism is implemented via multiplying a mask with corresponding attention domain. On the one hand, the mask can be adopted in the spatial domain to highlight the significant spatial area in feature maps [40]. On the other hand, attention mechanism can be applied in channel domain to focus on key feature channels [41]. Moreover, it is also practicable to introduce attention mechanism into both spatial and channel domain [42].

IV. From Memory to RNNs

This section mainly focuses on recurrent neural network (RNN) and how it is motivated from the perspective of information transmission.

A. Motivation on Information Transmission of RNN

Even the simplest human behavior can be broken down into a variety of serially ordered action sequences. Therefore, we need a model to process the sequential patterns of data like speech and context. An intuitive approach is to process temporal data spatially. However, one shortcoming of this kind of methods is that the duration of input patterns is required to be fixed, which breaks the completeness of temporal sequential information. Another problem is that, it is hard for such an approach to recognize absolute displacements of patterns with similar relative structure.

For this reason, Elman proposes a network architecture with a structure of recurrent connection that represents time implicitly [11]. This new type of neural network is named as recurrent neural network (RNN). The network will make an inference for the data of each time step. In addition, the recurrent connection constructs a path from the hidden state of the previous time step to the input of the current time step. Therefore, RNN can reserve the "memory" about the past information when making current inference. To some extend, this process actually fits the cognition mechanism of human brains: the understanding of an object mainly depends on our memory rather than the information we receive currently.

B. Details of Architecture

RNN takes as input one element of the sequential data at each time step. Meanwhile, as shown in Fig. 4(a), there is an additional layer of context neurons that can preserve the hidden state of the previous time step. At each time step t, the hidden layer will give a hidden state s_t through integrating the new information from input x_t and previous information, i.e., hidden state s_{t-1} (see Fig. 4(b)). This process can be formulated as:

$$s_t = h(U \cdot x_{t-1} + W \cdot s_{t-1} + b_h), \tag{2}$$

$$y_t = g(V \cdot s_t + b_g), \tag{3}$$

where b_h and b_g are biases, h denotes a hyperbolic tangent (*tanh*) function and g denotes a *softmax* function.

Due to that all functions that are adopted to obtain the actual output of RNN are differentiable, the network architecture can be trained using backpropagation [44]. The backpropagation algorithm for RNN is essentially equivalent to that for other network architectures. It is important to notice that the process of updating W and U is a little complex. For W, assuming that L_t is the error

generated at time step t. Then, the derivative of L_t with respect to W can be written as follows:

$$\frac{\partial L_t}{\partial W} = \frac{\partial L_t}{\partial y_t}\frac{\partial y_t}{\partial s_t}\frac{\partial s_t}{\partial W}. \tag{4}$$

Considering that $s_t = h(Ux_{t-i} + Ws_{t-1} + b_h)$ and s_{t-1} is also influenced by W and s_{t-2}. According to the chain rule, we have

$$\frac{\partial s_t}{\partial W} = \sum_{i=1}^{t} \frac{\partial s_t}{\partial s_i}\frac{\partial s_i}{\partial W}. \tag{5}$$

In addition, we also have

$$\frac{\partial s_t}{\partial s_i} = \prod_{k=i+1}^{t} \frac{\partial s_k}{\partial s_{k-1}}. \tag{6}$$

Therefore, $\partial L_t/\partial W$ can be formulated as

$$\frac{\partial L_t}{\partial W} = \sum_{i=1}^{t} \frac{\partial L_t}{\partial y_t}\frac{\partial y_t}{\partial s_t}\left(\prod_{k=i+1}^{t} \frac{\partial s_k}{\partial s_{k-1}}\right)\frac{\partial s_i}{\partial W}. \tag{7}$$

Similarly, the derivative of L_t with respect to U can be formulated as follows:

$$\frac{\partial L_t}{\partial U} = \sum_{i=1}^{t} \frac{\partial L_t}{\partial y_t}\frac{\partial y_t}{\partial s_t}\left(\prod_{k=i+1}^{t} \frac{\partial s_k}{\partial s_{k-1}}\right)\frac{\partial s_i}{\partial U}. \tag{8}$$

C. Evolution of RNNs

Despite the effectiveness in sequential data processing, RNN still suffers from gradient vanishing or explosion. This is because that there exists a lot of partial derivatives in model training (see Eq. 7), especially when the number of time steps is very large. If $(\partial s_k/\partial s_{k-1}) > 1$, then the gradient approaches infinity when the length of input time series is too large. In contrast, if $(\partial s_k/\partial s_{k-1}) < 1$. then the gradient will be close to 0. To solve this problem, adopting ReLu as activation function instead of tanh can slightly alleviate the vanishing gradient [27]. A better method is to change the propagation structure of RNNs to keep the partial derivatives equal to 1. In this case, the product of partial derivatives would no longer converge to 0 or diverge gradually. To satisfy this demand, Hochreiter and Schmidhuber [27] present a long short-term memory (LSTM) model with several "gate" structures. Inherently, a "gate" is a specific function. There are generally three kinds of gates: forget gate *for*, input gate *in* and output gate *out*. Then, the inference process of LSTM can be formulated as:

$$for(x_t, s_{t-1}) = \sigma(W_f \cdot conc(x_t, s_{t-1}) + b_f), \tag{9}$$

$$in(x_t, s_{t-1}) = \sigma(W_i \cdot conc(x_t, s_{t-1}) + b_i), \tag{10}$$

$$c'_t = h(W_c \cdot conc(x_t, s_{t-1}) + b_c), \tag{11}$$

$$c_t = for(x_t, s_{t-1}) * c_{t-1} + in(x_t, s_{t-1}) * c'_t, \tag{12}$$

$$out(x_t, s_{t-1}) = \sigma(W_o \cdot conc(x_t, s_{t-1}) + b_o), \tag{13}$$

$$s_t = out(x_t, s_{t-1}) * h(c_t), \tag{14}$$

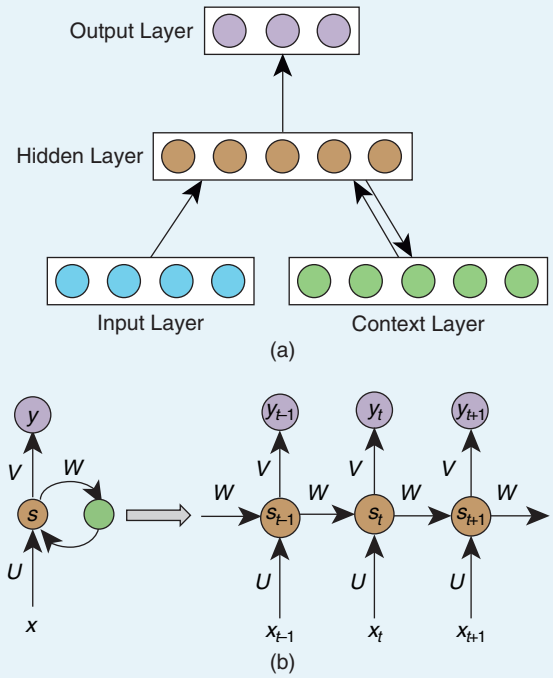


Figure 4. The illustration of RNN where U, V and W are respectively weights between input layer and hidden layer, weights between hidden layer and output layer and weights between hidden layer and context layer. (a) The structure of RNN [11], (b) The inference process of RNN [43].

where $conc(\cdot)$ denotes the concatenation operator and σ denotes sigmoid function. These "gates" can flexibly control the filtration of the information flow in the network. Thereby, the gradient problems of vanilla RNNs can be solved via making the product of the partial derivatives always equal to either 0 or 1 when computing the gradients.

Furthermore, many variants of the "gate" structures are suggested to improve the performance. Gers and Schmidhuber [45] enable the gates to observe the cell state c_{t-1} through some "peephole" connections. Cho et al. [46] simplify the transmission structure of information flow via integrating the forgot gate and input gate into one update gate. Moreover, hidden state and cell state are also merged in [46]. Lei et al. [47] propose a simple recurrent unit that can accelerate network computing via parallelizing the recurrence. Recently, another attempt to handle the gradient problem is to replace the weight matrices of RNN with the unitary matrix [48]. In addition, the attention mechanism is also introduced into the structure of RNNs to improve the flexibility and effectiveness in different application scenarios, such as natural language processing [49] and image captioning [50].

V. From Game Theory to GANs

This section mainly focuses on generative adversarial network (GAN) and how it is motivated from the perspective of learning strategy.

A. Motivation on Learning Strategy of GAN

Although classic generative models, such as hidden Markov models [51] and deep belief networks [12], have been successfully applied in various fields, they still have some inevitable limitations. Generally, generative models are based on the principle of maximum likelihood, that is to find a model that estimates a probability distribution which approximates the distribution of the real data [52]. From this point of view, defining a probability density explicitly to describe the data distribution is convenient due to the tractability of given density function in the computation of maximum likelihood estimation. However, it is hard for the defined probability density to capture the characteristics of real data perfectly and represent the complexity of high-dimensional data distribution [53].

To avoid the drawbacks mentioned above, another approach is to represent the probability density implicitly in a data-driven way. Inspired by the theory of zero-sum game, a novel learning strategy, termed adversarial learning, is developed to help neural network learn a probability distribution which approximates the real data distribution [13]. The network framework corresponding to adversarial learning is generative adversarial networks (GAN). Through adversarial learning, two neural networks (generator and discriminator) in GAN are optimized with two opposite objectives: one is responsible for learning to generate samples to deceive the other while the other is trained not to be deceived. According to game theory, the competition between the two networks can reach Nash equilibrium at last, means the data distribution is learned by generator network successfully [52].

B. Details of Architecture

As mentioned above, GAN consists of two competing sub-models: the generative model G (i.e. generator) and the discriminative model D (i.e. discriminator). Technically, GAN is rather a framework than a neural network. Like the encoder-decoder framework, the two sub-models of GAN can be constructed flexibly with various neural network models. In vanilla GAN [13], the generator and discriminator are both multi-layer perceptrons.

The learning process of GAN is modeled as a game (see Fig. 5). In this game, G is in charge of yielding a sample $G(\mathbf{z})$ according to a stochastic noise $\mathbf{z} \sim p_{\text{noise}}$ to deceive the discriminator, where p_{noise} is the distribution of the stochastic noise. Then, D differentiates the sample $G(\mathbf{z})$ from a real sample $\mathbf{x} \sim p_{\text{data}}$, where p_{data} is the distribution of the real dataset. Specifically, the discriminator D takes as input both the fake sample $G(\mathbf{z})$ and a real sample $\mathbf{x} \sim p_{\text{data}}$. For $G(\mathbf{z})$, D gives 0 as a negative response. While for $\mathbf{x} \sim p_{\text{data}}$, D outputs 1 as a positive response. It can be proved that when the generator and discriminator are trained sufficiently, the model G can learn a distribution that approaches the real data distribution p_{data} [13]. In this case, both $D(\mathbf{x})$ and $D(G((z)))$ are close to 0.5, which means D can no longer recognize which sample is real. This process can be summarized as a minimax game between D and G and the objective function $O(D, G)$ is:

$$\min_G \max_D O(D,G) = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim p_{\text{noise}}}[\log(1 - D(G(\mathbf{z})))], \tag{15}$$

or

$$\min_G \max_D O(D,G) = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim p_{\text{noise}}}[-\log(D(G(\mathbf{z})))]. \tag{16}$$

The GAN framework is innovative as it replaces the variational lower bounds or Markov chains with supervised learning to approximate the distribution of real data

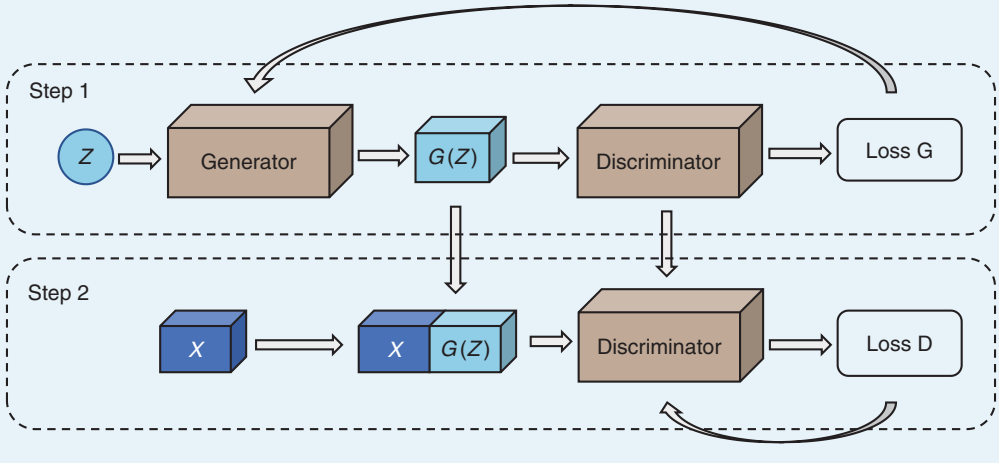


Figure 5. The training process of GAN can be divided into two iterative steps: 1) fixing the parameters of the discriminator and training the generator; 2) training the discriminator using a real sample **x** and a generated sample $G(\mathbf{Z})$.

[52]. However, the adversarial learning also results in new problems. For instance, the training of GAN requires G and D being optimized iteratively, which is time-consuming. Moreover, if we adopt the objective function in Eq. 15, the training objective is equivalent to minimizing the Jensen-Shannon (JS) divergence between the real data distribution p_{data} and the generator distribution p_G. While if we adopt the objective function in Eq. 16, then the training objective is to minimize the Kullback-Leibler (KL) divergence between p_{data} and p_G and maximize the JS divergence between the two distributions simultaneously. In the first case, using JS divergence as the loss function can lead to gradient vanishing because when the two distributions have significant difference, the JS divergence is close to a constant. In the second case, due to that the KL divergence is asymmetric, the generator prefers to generate samples with fixed patterns and ignore the sample diversity [54].

C. Evolution of GANs

To fix such theoretical flaws, a mathematical tool is introduced into the framework of GAN. Wasserstein GAN [55] leverages *Wasserstein distance* to substitute the KS and JS divergence as the measurement of the distribution distance. This distance measurement is symmetric and can reflect the distribution distance effectively even in case that two distributions gap largely. That is to say, the Wasserstein distance can fundamentally solve the problems of gradient vanishing as well as diversity lacking in GAN. Besides, another idea to enhance the model stability is to raise the resolution of synthesized images progressively. According to this, a progressive training mechanism is developed in ProGAN [56], which trains the generator and discriminator layer by layer.

In image generation, the impressive performance of GAN attracts a lot of attention and plenty of variants are presented in recent years. Due to the high flexibility of adversarial framework, GAN can be combined with CNN to leverage its ability of image processing and produce images with better quality [57]. Moreover, variational autoencoder can also be adopted as the generator in GAN framework, which is less susceptible to the problem of model collapse [58].

Besides, label information can be introduced into the input space of GAN to generate samples with specified type. Conditional GAN [59] decomposes the input space of GAN into stochastic noise z and class label y. In this case, the generated image can be modified by manipulating in the latent space rather than directly in the image space where the modification is difficult because the image distributions lie in high-dimensional complex manifolds.

In addition, the appearance of GANs leads to better solutions to a problem called style translation or image-to-image translation. For two different image styles, two GANs are trained in cycleGAN [60]. One GAN translates the image from the original domain to the objective domain and the other GAN translates the image from the objective domain to the original domain. Except for the standard training of GAN, an additional cycle consistency loss is introduced to reduce the discrepancy between the input image x and the image generated via bi-directional translation from x.

In recent years, attention mechanism is widely applied in RNNs, CNNs and other deep neural architectures to allocate more attention to meaningful information. By adopting self-attention mechanism, SAGAN [61] learns an attention map from the feature maps to model the non-local dependencies between feature regions. Thus, SAGAN can generate details of images considering the information from farther location in the feature maps.

VI. Conclusion and Prospect

With continuously deepening of the research about DNNs, various deep neural architectures have been presented and widely applied in many fields. In this paper, we have reviewed the intrinsic motivation of some widely-used deep neural architectures. These motivations have promoted the development and innovation of existing neural architectures from different aspects. Using this as a starting point, we have roughly categorized the influence domains of motivations for some DNNs as three classes, namely information processing, information transmission and learning strategy. To further illustrate how neural architectures are motivated and developed from the three aspects, convolutional neural network, recurrent neural network and generative adversarial network have been taken as instances and discussed respectively in detail. According to the discussion, some conclusions have been summarized as follows.

In the aspect of information processing, deep neural architectures can be developed by improving the working mechanism of neurons or adjusting the information directly. On the one hand, the ability of feature capturing can be enhanced by changing the sample mechanism of neurons. For example, sampling one-dimensional vector globally is transformed to sampling two-dimensional plane locally in CNNs. On the other hand, representations can be modified by a learned mask based on attention mechanism.

In the aspect of information transmission, network motivations focus on the utilization of features through changing the path structure of information flow in DNNs. Appropriate path connection cannot only improve the network's ability of representation learning, but also achieve the embedding of special data such as time series and graphs.

In the aspect of learning strategy, according to specific task requirements, the proposal of a novel learning strategy can be realized by presenting a corresponding

objective function for network training. Meanwhile, the realization of such a learning strategy generally needs a matching network framework, such as the encoder-decoder framework for self-supervised learning and GAN framework for adversarial learning.

In order to grasp the future trend, some promising research topics about deep neural architectures are also discussed in this paper.

- **Neural networks based on brain mechanisms:** The architecture of DNNs can be regarded as an imitation of the hierarchical structure of human speech perception and production systems [62]. In the case of convolutional neural networks, the raw image information is transformed into a more abstract representation layer by layer until final cognition is formed. With the deepening understanding of the brain nerve structure and information processing mechanism, novel neural networks based on brain mechanisms are showing a level of intelligence closer to that of human. At present, a typical brain-like neural network is spiking neural network (SNN) [63], [64]. The model of SNN is constructed to be closer to the working mechanism of biological neural networks. Unlike traditional neural networks, the information carrier processed by neurons in SNN is impulse train. This way of information processing is in fact a simulation of the accumulation of membrane potential and impulse discharge of biological neurons after reaching the threshold. These bionic characteristics lead to stronger capability of complex information processing. However, it is hard to train a SNN due to the non-differentiability of neurons. Thus, a general learning strategy for such a novel neural architecture is a significant research direction.
- **Automatically designing network architecture:** Deep neural networks have attained outperforming achievement in the automation of feature extraction, but they still suffer from the high price to acquire a good network architecture. Designing an appropriate network aiming at specified task generally requires a large amount of time prior expertise for trial and exploration. Thus, how to efficiently search an available network architecture in low cost is destined to be a research hotspot in the future. Neural Architecture Search (NAS) is an area of greatest interest in automating machine learning. This technique is used to automatically design network architecture by exploring network structures and hyperparameters in specified search space with some strategies. Elsken et al. [65] summarize existed researches in this field and analyze these works from three aspects: search space, search strategy and performance estimation strategy. At present, most studies of NAS are conducted for image classification. More architectures for other domains, such as image restoration, semantic segmentation and natural language processing, are waiting to be further explored.
- **Training neural network via contrastive learning:** GAN is essentially a generative model that needs to learn to construct as more sample details as possible. But sometimes, the network only needs to learn a representation which can used to distinguish different samples. The learning strategy motivated by this idea is contrastive learning. For a particular sample x, we can construct some positive samples $\{x^+\}$ and negative samples $\{x^-\}$ through a certain transformation. The objective of contrastive learning is to maximize the consistency between $f(x)$ and $f(x^+)$ while minimize that between $f(x)$ and $f(x^-)$, where f is a mapping function. Related mechanism and framework of contrastive learning can be seen in [66], [67]. About contrastive learning, there are two key questions to be answered. The first one is how to define a metric to measure the difference between the representation of samples. An effective metric is indispensable to an appropriate objective function for contrastive learning. The second one is how to construct the transformation to generate appropriate positive and negative samples. For data from different domains, such as image, text and audio signal, it is important to find corresponding transformation that can highlight the semantic relation between sample pair.

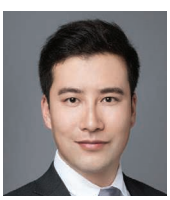
Acknowledgments

This work was supported in part by the National Key Research and Development Program of China under Grants 2018AAA0101100, 2016YFB0800401, in part by the National Natural Science Foundation of China under Grants 61621003, 61903017, 61532020, and in part by the China Postdoctoral Science Foundation under Grant 2020M670087.

Weilin Luo received his B.S. degree in information and computing science and M.S. degree in automation science from Nanjing University of Aeronautics and Astronautics, Nanjing, China in 2018. He is currently pursuing the Ph.D. degree with the school of Automation science and Electrical Engineering, Beihang University, Beijing, China. His current research interests include intellisense and intelligent decision making techniques based on deep neural architectures.

Jinhu Lü (M'03-SM'06-F'13) received the Ph.D. degree in applied mathematics from the Academy of Mathematics and Systems Science (AMSS), Chinese Academy of Sciences, Beijing, China, in 2002. Currently, he is the Dean with the School of Automation Science and Electrical Engineering, Beihang University, Beijing, China. Also, he is a Professor with the AMSS, Chinese Academy of Sciences. He was a Professor with RMIT University, Melbourne, VIC, Australia, and a Visiting Fellow with Princeton University, Princeton, NJ, USA. He is a Chief Scientist of National Key Research and Development Program of China and a Leading Scientist of Innovative Research Groups of NNSF of China. His current research interests include complex networks, industrial Internet, network dynamics and cooperation control. Dr. Lü was a recipient of the prestigious Ho Leung Ho Lee Foundation Award in 2015, the National Innovation Competition Award in 2020, the State Natural Science Award three times from the Chinese Government in 2008, 2012, and 2016, respectively, the Australian Research Council Future Fellowships Award in 2009, the NNSF of Distinguished Young Scholars Award, and the Highly Cited Researcher Award in engineering from 2014 to 2019. He is/was an Editor in various ranks for 15 SCI journals, including the Co-Editor-in-Chief of IEEE TII. He served as a member in the Fellows Evaluating Committee of the IEEE CASS, the IEEE CIS, and the IEEE IES. He was the General Co-Chair of IECON 2017. He is the Fellow of IEEE and CAA.

Xuerong Li received the B.E. degree from Renmin University, China in 2013 and the Ph.D. degree from the University of Chinese Academy of Sciences in 2019. She was an Assistant Professor with the Academy of Mathematics and Systems Science, Chinese Academy of Sciences. Her research interests include complex networks, machine learning, and economic forecasting.



Lei Chen received the Ph.D. degree in control theory and engineering from Southeast University, Nanjing, China, in 2018. He was a visiting Ph.D. student with the RMIT University, Melbourne, VIC, Australia, and the Okayama Prefectural University, Soja, Japan. Currently, he is a Post-Doctoral Fellow with the School of Automation Science and Electrical Engineering, Beihang University, Beijing, China. His current research interests include complex networks, characteristic modeling approach, spacecraft control, and network control.

Kexin Liu received the M.Sc. degree in control science and engineering from Shandong University, Jinan, China in 2013, and Ph.D degree in System Theory from Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, China in 2016, respectively. From 2016 to 2018, he was a Post-Doctoral Fellow with Peking University, Beijing. Currently, he is an Associated Professor with the School of Automation Science and Electrical Engineering, Beihang University, Beijing, China. His research interests include multi-agent systems and complex networks.

References

[1] M. Zhang, J. Lü, Z. Bai, H. Liu, and C. Fan, "Improving the initialization speed for long-range NRTK in network solution mode," *Sci. China Technol. Sci.*, vol. 63, no. 5, pp. 866–873, May 2020. doi: 10.1007/s11431-019-9507-8.

[2] M. Zhang, J. Lü, Z. Bai, Z. Jiang, and B. Chen, "An overview on GNSS carrier-phase time transfer research," *Sci. China Technol. Sci.*, vol. 62, no. 8, pp. 1412–1422, Apr. 2020. doi: 10.1007/s11431-019-9655-1.

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015. doi: 10.1038/nature14539.

[4] A. Ivakhnenko and V. G. Lapa, *Cybernetic Predicting Devices*. New York: CCM Information Corp., 1965.

[5] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *J. Physiol.*, vol. 160, no. 1, pp. 106–154, Jan. 1962. doi: 10.1113/jphysiol.1962.sp006837.

[6] K. Fukushima, "Neural network model for a mechanism of pattern recognition unaffected by shift in position-neocognitron," *IEICE Tech. Rep. A*, vol. 62, no. 10, pp. 658–665, 1979.

[7] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998. doi: 10.1109/5.726791.

[8] K. Fukushima, "Training multi-layered neural network neocognitron," *Neural Netw.*, vol. 40, pp. 18–31, Jan. 2013. doi: 10.1016/j.neunet.2013.01.001.

[9] H. Bourlard and Y. Kamp, "Auto-association by multilayer perceptrons and singular value decomposition," *Biol. Cybern.*, vol. 59, nos. 4–5, pp. 291–294, Sep. 1988. doi: 10.1007/BF00332918.

[10] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015. doi: 10.1016/j.neunet.2014.09.003.

[11] J. L. Elman, "Finding structure in time," *Cogn. Sci.*, vol. 14, no. 2, pp. 179–211, Apr. 1990. doi: 10.1207/s15516709cog1402_1.

[12] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, July 2006. doi: 10.1162/neco.2006.18.7.1527.

[13] I. Goodfellow et al., "Generative adversarial nets," in *Proc. 27th Adv. Neural Inf. Process. Syst. (NIPS)*, 2014, pp. 2672–2680. doi: 10.5555/2969033.2969125.

[14] L. Tang, J. Lu, and J. Lü, "A threshold effect of coupling delays on intra-layer synchronization in duplex networks," *Sci. China Technol. Sci.*, vol. 61, no. 12, pp. 1907–1914, Dec. 2018. doi: 10.1007/s11431-017-9285-7.

[15] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Proc. 30th Adv. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 3856–3866.

[16] W. Sultani, C. Chen, and M. Shah, "Real-world anomaly detection in surveillance videos," in *Proc. 2018 IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, June 2018, pp. 6479–6488.

[17] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," *ACM Comput. Surv. (CSUR)*, vol. 52, no. 1, pp. 1–38, Feb. 2019. doi: 10.1145/3285029.

[18] B. Li, B. Hou, W. Yu, X. Lu, and C. Yang, "Applications of artificial intelligence in intelligent manufacturing: A review," *Front. Inf. Technol. Electron. Eng.*, vol. 18, no. 1, pp. 86–96, Feb. 2017. doi: 10.1631/FITEE.1601885.

[19] H. Gu, J. Lü, and Z. Lin, "On PID control for synchronization of complex dynamical network with delayed nodes," *Sci. China Technol. Sci.*, vol. 62, no. 8, pp. 1412–1422, Aug. 2019. doi: 10.1007/s11431-018-9379-8.

[20] L. Wu, P. Wang, and J. Lü, "Substrate concentration effect on gene expression in genetic circuits with additional positive feedback," *Sci. China Technol. Sci.*, vol. 61, no. 8, pp. 1175–1183, Aug. 2018. doi: 10.1007/s11431-018-9301-0.
[21] R. F. Molanes, K. Amarasinghe, J. Rodriguez-Andina, and M. Manic, "Deep learning and reconfigurable platforms in the internet of things: Challenges and opportunities in algorithms and hardware," *IEEE Ind. Electron. Mag.*, vol. 12, no. 2, pp. 36–49, Jun. 2018. doi: 10.1109/MIE.2018.2824843.
[22] M. Hutson, "Core progress in AI has stalled in some fields," *Science*, vol. 368, no. 6494, pp. 927–927, May 2020.
[23] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," 2017. [Online]. Available: https://arxiv.xilesou.top/abs/1706.05587
[24] J. Dai et al., "Deformable convolutional networks," in *Proc. 2017 IEEE Int. Conf. Comput. Vis. (ICCV)*, pp. 764–773.
[25] C. Szegedy et al., "Going deeper with convolutions," in *Proc. 2015 IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 1–9.
[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. 2016 IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 770–778.
[27] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. doi: 10.1162/neco.1997.9.8.1735.
[28] X. Wang, H. Gu, Q. Wang, and J. Lü, "Identifying topologies and system parameters of uncertain time-varying delayed complex networks," *Sci. China Technol. Sci.*, vol. 62, no. 1, pp. 94–105, Jan. 2019. doi: 10.1007/s11431-018-9287-0.
[29] K. Cheng, Y. Chen, and W. Fang, "Improved object detection with iterative localization refinement in convolutional neural networks," *IEEE Trans. Circuits Syst. Vid. Technol.*, vol. 28, no. 9, pp. 2261–2275, July 2018. doi: 10.1109/TCSVT.2017.2730258.
[30] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, "Fully-convolutional Siamese networks for object tracking," in *Proc. 2016 Eur. Conf. Comput. Vis. (ECCV)*, pp. 850–865.
[31] J. Wang, W. Wang, and W. Gao, "Multiscale deep alternative neural network for large-scale video classification," *IEEE Trans. Multimedia*, vol. 20, no. 10, pp. 2578–2592, July 2018. doi: 10.1109/TMM.2018.2855081.
[32] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," 2018. [Online]. Available: https://arxiv.org/abs/1603.07285
[33] B. M. Wilamowski, "Neural network architectures and learning algorithms," *IEEE Ind. Electron. Mag.*, vol. 3, no. 4, pp. 56–63, Dec. 2009. doi: 10.1109/MIE.2009.934790.
[34] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. 14th Int. Conf. Artif. Intell. Stat. (ICAIS)*, 2011, pp. 315–323.
[35] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, 2010, pp. 807–814. doi: 10.5555/3104322.3104425.
[36] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 4700–4708.
[37] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. 25th Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105. doi: 10.1145/3065386.
[38] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. 2018 IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 6848–6856. doi: 10.1109/CVPR.2018.00716.
[39] M. Wang, B. Liu, and H. Foroosh, "Factorized convolutional neural networks," in *Proc. 2017 IEEE Int. Conf. Comput. Vis. Workshops (ICCV)*, pp. 545–553. doi: 10.1109/ICCVW.2017.71.
[40] S. Woo, J. Park, J. Lee, and I. S. Kweon, "CBAM: Convolutional block attention module," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sept. 2018, pp. 3–19. doi: 10.1007/978-3-030-01234-2_1.
[41] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. 2018 IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 7132–7141. doi: 10.1109/CVPR.2018.00745.
[42] F. Wang et al., "Residual attention network for image classification," in *Proc. 2017 IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 6450–6458. doi: 10.1109/CVPR.2017.683.
[43] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A critical review of recurrent neural networks for sequence learning," 2015. [Online]. Available: https://arxiv.xilesou.top/abs/1506.00019
[44] M. Boden, "A guide to recurrent neural networks and backpropagation," *In the Dallas project*, 2002.
[45] F. A. Gers and J. Schmidhuber, "Recurrent nets that time and count," in *Proc. IEEE-INNS-ENNS Int. Joint Conf. Neural Netw. (IJCNN)*, vol. 3, 2000, pp. 189–194. doi: 10.1109/IJCNN.2000.861302.
[46] K. Cho et al., "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proc. 2014 Conf. Empir. Methods Nat. Lang. Process. (EMNLP)*, pp. 1724–1734. doi: 10.3115/v1/D14-1179.
[47] T. Lei, Y. Zhang, S. I. Wang, H. Dai, and Y. Artzi, "Simple recurrent units for highly parallelizable recurrence," in *Proc. 2018 Conf. Empir. Methods Nat. Lang. Process. (EMNLP)*, pp. 4470–4481.
[48] M. Arjovsky, A. Shah, and Y. Bengio, "Unitary evolution recurrent neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1120–1128. doi: 10.5555/3045390.3045509.
[49] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014. [Online]. Available: https://arxiv.org/abs/1409.0473
[50] K. Xu et al., "Show, attend and tell: Neural image caption generation with visual attention," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2015, pp. 2048–2057. doi: 10.5555/3045118.3045336.
[51] S. R. Eddy, "Hidden Markov models," *Curr. Opin. Struct. Biol.*, vol. 6, no. 3, pp. 361–365, June 1996. doi: 10.1016/S0959-440X(96)80056-X.
[52] I. Goodfellow, "NIPS 2016 tutorial: Generative adversarial networks," 2016. [Online]. Available: https://arxiv.xilesou.top/abs/1701.00160
[53] A. Nguyen, A. Dosovitskiy, J. Yosinski, T. Brox, and J. Clune, "Synthesizing the preferred inputs for neurons in neural networks via deep generator networks," in *Proc. 29th Adv. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 3387–3395. doi: 10.5555/3157382.3157477.
[54] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," 2017. [Online]. Available: https://arxiv.org/abs/1701.04862
[55] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 214–223.
[56] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation," in *Proc. 6th Int. Conf. Learn. Represent. (ICLR)*, 2018.
[57] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015. [Online]. Available: https://arxiv.xilesou.top/abs/1511.06434
[58] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther, "Autoencoding beyond pixels using a learned similarity metric," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1558–1566.
[59] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014. [Online]. Available: https://arxiv.xilesou.top/abs/1411.1784
[60] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. IEEE Int. Conf Comput. Vis. (ICCV)*, 2017, pp. 2223–2232. doi: 10.1109/ICCV.2017.244.
[61] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 7354–7363.
[62] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, Apr. 2017. doi: 10.1016/j.neucom.2016.12.038.
[63] A. Shukla and U. Ganguly, "An on-chip trainable and the clockless spiking neural network with 1R memristive synapses," *IEEE Trans. Biomed. Circuits Syst.*, vol. 12, no. 4, pp. 884–893, May 2018. doi: 10.1109/TBCAS.2018.2831618.
[64] T. Zhang, Y. Zeng, D. Zhao, and M. Shi, "A plasticity-centric approach to train the non-differential spiking neural networks," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018.
[65] S. Ivanov and A. D'yakonov, "Modern deep reinforcement learning algorithms," 2019. [Online]. Available: https://arxiv.xilesou.top/abs/1906.10025
[66] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, June 2020, pp. 9726–9735. doi: 10.1109/CVPR42600.2020.00975.
[67] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," 2020. [Online]. Available: https://arxiv.org/abs/2002.05709