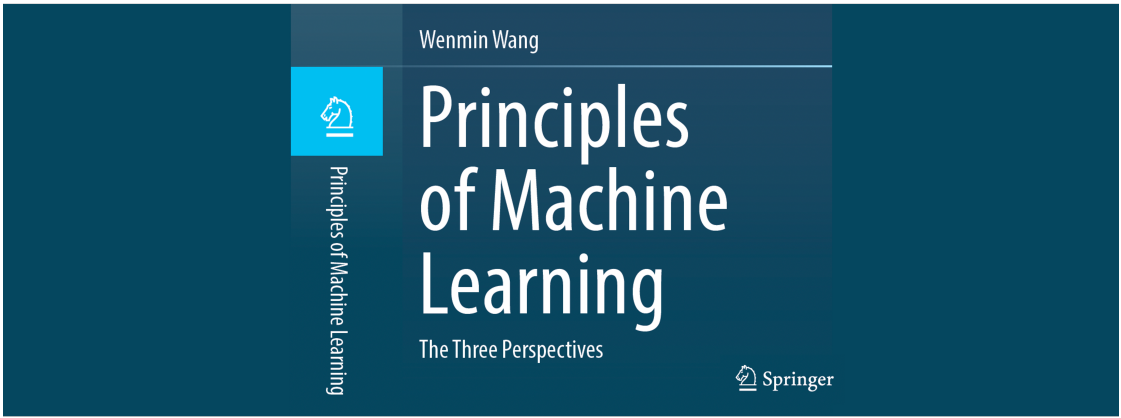


https://link.springer.com/book/10.1007/978-981-97-5333-8

Part II Frameworks

5 Connectionist Framework

5 Connectionist Framework

About the Connectionist Framework

- The framework is based on connectionist approaches. Its manifestation is artificial neural networks (ANNs), used to simulate the brain's information processing and learning process.
- ANNs were pioneered in 1940s, gradually becoming a theory and method in AI and machine learning.
- Hebbian theory is an interpretation of the brain's learning process, and is therefore also known as Hebbian learning.
- Connectionist core is the theory of connectionist learning based on ANNs.

5 Connectionist Framework

Connectionist Learning Theories

History Related to Connectionist Learning Theories

Year	Name	Pioneer(s) and Publication
1949	Hebbian Theory	Donald Hebb. *The Organization of Behaviour: A Neuropsychological Theory*, Jone Wiley & Sons, 1949.
1982	Connectionist Models	Jerome Feldman and Dana Ballard. "Connectionist Models and Their Properties", *Cognitive Science*, 1982.
1986	Parallel Distributed Processing	David Rumelhart, Geoffrey Hinton and James McClelland. "A general framework for parallel distributed processing." *Parallel distributed processing: Explorations in the microstructure of cognition*, 1986.
2007	Neural Networks Theory	Alexander I. Galushkin. *Neural networks theory*. Springer Berlin Heidelberg, 2007.

Hebbian Theory

Hebbian theory was introduced by Donald Hebb in his book The Organization of Behavior: A Neuropsychological Theory, in 1949.

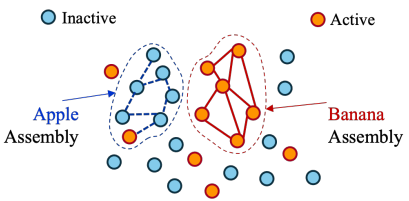
Hebb attempts to explain the adaptation of brain neurons during the learning process.

> "When an axon of cell *A* is near enough to excite a cell *B* and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that *A*'s efficiency, as one of the cells firing *B*, is increased."

It is also called Hebb's rule, and cell-assembly theory.

Hebb's Cell-Assembly Theory

"Any frequently repeated, particular stimulation will lead to the slow development of a 'cell-assembly,' a diffuse structure comprising cells in the cortex and diencephalon (and also, perhaps in the basal ganglia of the cerebrum), capable of acting briefly as a closed system, delivering facilitation to other such systems and usually having a specific motor facilitation."



"A series of such events constitutes a 'phase sequence' – the thought process. Each assembly action may be amused by a preceding assembly, by a sensory event, or-normally-by both. The central facilitation from one of these activities on the next is the prototype of 'attention'."

Connectionist Models

Connectionism attempts to explain mental phenomena using ANNs. Inspired by the brain, using neuron-like units, and synapse-like connections.

Connectionism also proposes a cognitive theory, based on simulating concurrent distributed signal activity through digitizable representations of connections, and the function of learning can be achieved by adjusting the weights of the connections.

Connectionist models was proposed by Jerome Feldman and Dana Ballard in 1982, which are designed as an information processing models (IPM).

They also discussed the stability and noise sensitivity of the model, distributed decision-making, issues of time and sequence, representation of complex concepts, and application in cognitive science.

Parallel Distributed Processing

Parallel Distributed Processing (PDP) began in the 1970s and made significant progress in the 1980s.

PDP emphasizes the parallel processing nature of neural networks and the distributed nature of neurons.

It became popular in 1986, and 1987 with the release of the books:

Parallel Distributed Processing: Explorations in the Microstructure of Cognition.

> *Volume 1: Foundations*
> *Volume 2: Psychological and Biological Models*.

Those books are now considered seminal connectionist works, and it is now common to fully equate PDP and connectionism.

Framework of Parallel Distributed Processing

In the framework for PDP, the *eight important attributes* were listed:

(1) A set of processing units.
(2) A state of activation.
(3) An output function for each unit.
(4) A pattern of connectivity among units.
(5) A propagation rule for propagating patterns of activities through the network.
(6) An activation rule for combining the inputs impinging on a unit with the current state of that unit to produce a new level of activation for the unit.
(7) A learning rule whereby patterns of connectivity are modified by experience.
(8) An environment within which the system must operate.

Neural Network Theory

The concept of neural networks can be traced back to the McCulloch-Pitts Neuron Model, proposed in 1943.

The perceptron, composed of single-layer neurons, is the earliest artificial neural network.

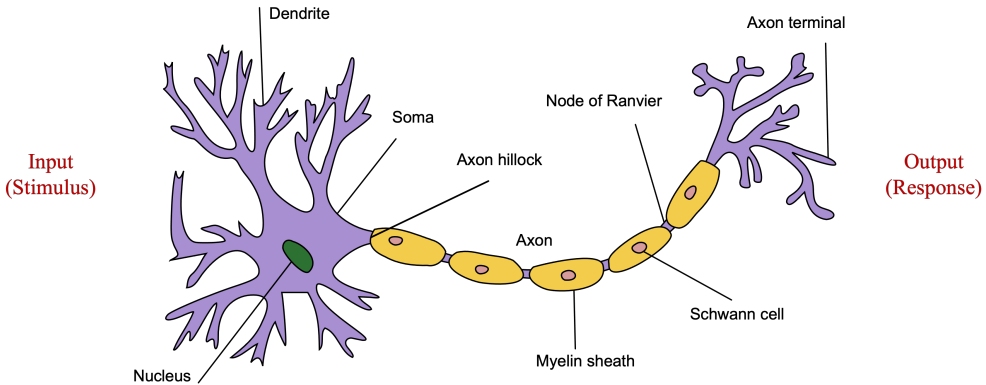
Multilayer perceptron (MLP) consists of an input layer, an output layer, and multiple hidden layers, also known as a multilayer artificial neural network.

In 1997, Philippe de Wilde published a book titled *Neural Network Models: Theory and Projects*, which deeply explained neural network models.

In 2007, Alexander Galushkin published a book titled *Neural Networks Theory*, elevating neural networks to a theoretical level.

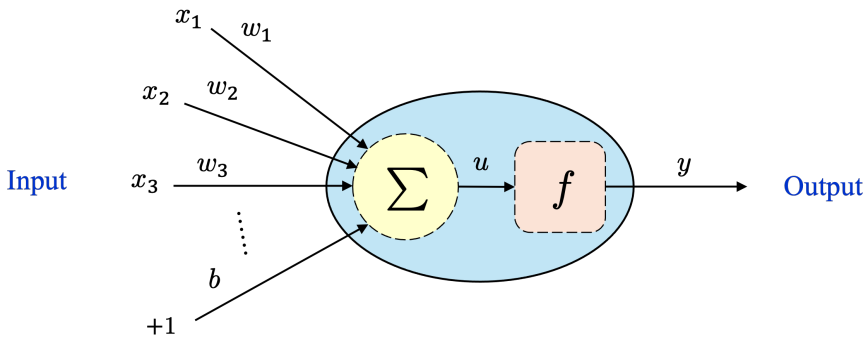
5 Connectionist Framework

Biological Neuron



https://en.wikipedia.org/wiki/Neuron

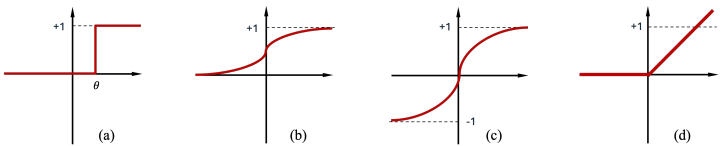
Artificial Neuron



$$u = \sum_i w_i x_i + b, \qquad \text{activation function} y = f(u).$$

Activation Functions

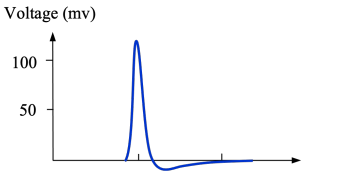
Name	Function	Derivative
(a) Binary threshold	$f(u) = \begin{cases} 0 & \text{if } u < \theta \\ 1 & \text{otherwise} \end{cases}$	$\frac{d\sigma(u)}{du} = 0$
(b) Sigmoid	$f(u) = \sigma(u) = \frac{1}{1+e^{-u}}$	$\frac{d\sigma(u)}{du} = \sigma(u) \cdot (1 - \sigma(u))$
(c) Hyperbolic Tangent (TanH)	$f(u) = \tanh(u) = \frac{(e^u - e^{-u})}{(e^u + e^{-u})}$	$\frac{d\tanh(u)}{du} = 1 - f(u)^2$
(d) Rectified Linear Unit (ReLU)	$f(u) = \max(0, u) = \begin{cases} 0, & u \leq 0 \\ u, & u > 0 \end{cases}$	$\frac{df(u)}{du} = \begin{cases} 0, & u \leq 0 \\ 1, & u > 0 \end{cases}$



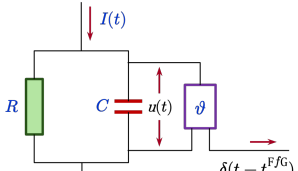
Spiking Neuron

Spike neuron was proposed in 1952, used to represent the pulse signal received by dendrites of biological neurons, its waveform is extremely sharp.

- It transmits information through a discrete spike train.
- Generating spikes increases/decreases with excitatory/inhibitory input.
- When internal state reaches a given threshold, spikes will be generated.



(a)



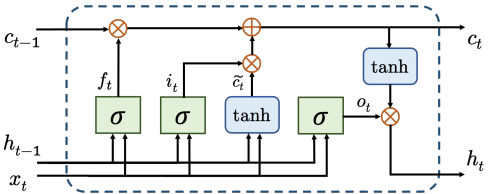
(b)

(a) Action potential.

(b) Leaky integrate-and-fire model.

Long Short-Term Memory Unit

Long short-term memory (LSTM) unit was proposed in 1997, which can serve as a unit of a recurrent neural network (RNN).

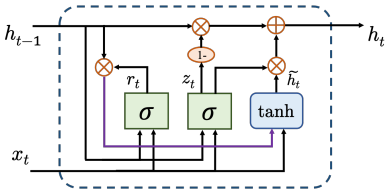


$$
\begin{aligned}
f_t &= \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right), \\
i_t &= \sigma\left(W_i x_i + U_i h_{t-1} + b_i\right), \\
\widetilde{c}_t &= \tanh\left(W_c x_t + U_c h_{t-1} + b_c\right), \\
c_t &= f_t * c_{t-1} + i_t * \widetilde{c}_t, \\
o_t &= \sigma\left(W_o x_t + U_o h_{t-1} + b_o\right), \\
h_t &= o_t * \tanh\left(c_t\right).
\end{aligned}
$$

A neural network composed of LSTM units is called an LSTM network.

Gated Recurrent Unit

Gated recurrent unit (GRU) is a variant of LSTM unit, proposed in 2014, can also serve as a unit of RNN.



$$
\begin{aligned}
z_t &= \sigma\left(W_z \cdot [h_{t-1}, x_t]\right), \\
r_t &= \sigma\left(W_r \cdot [h_{t-1}, x_t]\right), \\
\widetilde{h}_t &= \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right), \\
h_t &= (1 - z_t) * h_{t-1} + z_t * \widetilde{h}_t.
\end{aligned}
$$

GRU is more concise and performs similarly to LSTM in some tasks, and shows better performance on some smaller datasets.

But some studies suggest that LSTM is more powerful than GRU.

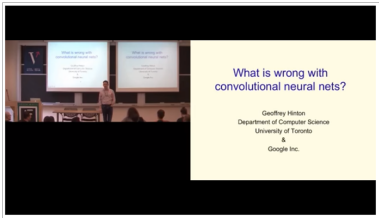
Capsule Unit

Geoffrey Hinton had a presentation titled “What is wrong with convolutional neural nets?” on August 20, 2017.

CNNs have too few levels of structure: neurons, layers, and whole nets.

It needs to group the neurons in each layer into “capsules”.

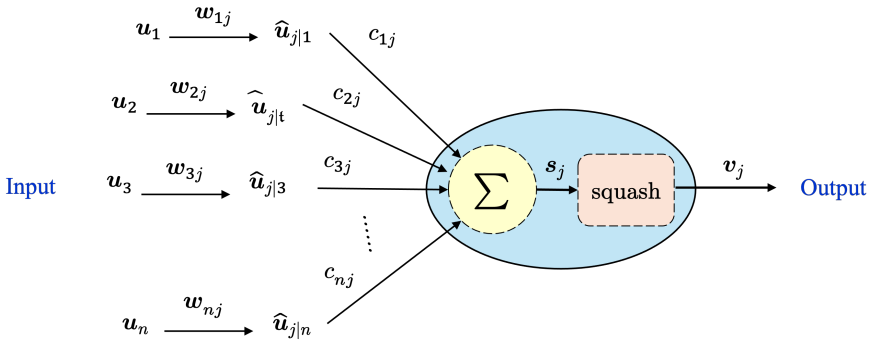
Capsules can do a lot of internal computation and output a compact result.



https://www.youtube.com/watch?v=Jv1VDdI4vy4

The network consisting capsules is called capsule neural network (CapsNet).

Capsule Unit

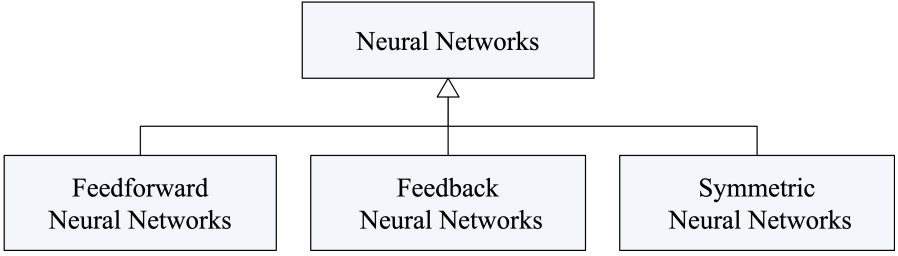


$$\hat{u}_{j|i} = w_{ij} u_i. \qquad s_j = \sum_i c_{ij} \hat{u}_{j|i}. \qquad .v_j = \frac{\| s_j \|^2}{1 + \| s_j \|^2} \frac{s_j}{\| s_j \|}.$$

5 Connectionist Framework

About the Structures of Neural Networks

From the viewpoints on the structures of neural networks, which can be divided into three main types:



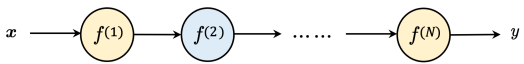
Feedforward Neural Networks

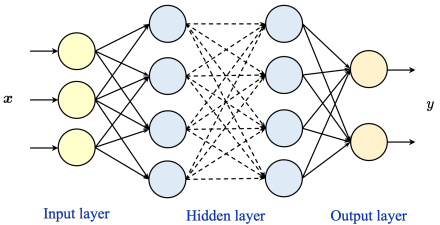
In feedforward neural networks, the information moves in only one direction, forward from input layer, through hidden layers and to the output layer.

A feedforward model with N layers can be represented by

$$y = f^{(N)} \circ \cdots \circ f^{(2)} \circ f^{(1)} \left(\boldsymbol{x}; \theta^{(1)}\right).$$

where $f^{(i)} \left(\boldsymbol{x}; \theta^{(i)}\right)$ is the function of ith layer.



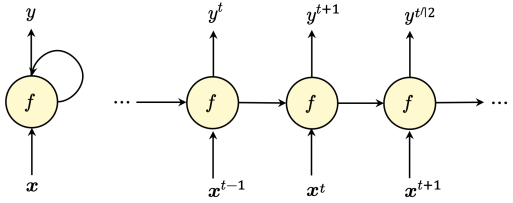


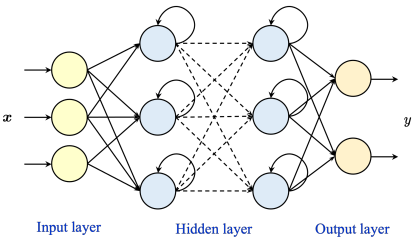
It's a multi-layer feedforward neural network if using artificial neural network to implement the multi-layer feedforward model.

Recurrent Neural Networks

Recurrent neural networks (RNNs) also include backward connections between their neurons or layers, forming directed cycles.

RNNs can retain internal state to show dynamic temporal characteristics.





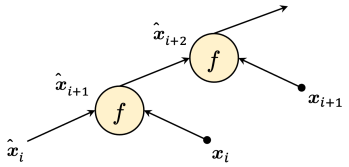
$$y^{(t+1)} = f\left(y^{(t)}, \boldsymbol{x}^{(t)}; \theta\right) = f\left(f\left(y^{(t-1)}, \boldsymbol{x}^{(t-1)}\right), \boldsymbol{x}^{(t)}; \theta\right); \theta).$$

Recursive Neural Networks

Recursive neural networks are a type of tree-like network.

Applying the same set of weights recursively and traverses the network in topological order.

$$\hat{\boldsymbol{x}}_{i+2} = f\left(\hat{\boldsymbol{x}}_{i+1}, \boldsymbol{x}_{i+1}; \theta\right) = f\left(f\left(\hat{\boldsymbol{x}}_i, \boldsymbol{x}_i; \theta\right), \boldsymbol{x}_{i+1}; \theta\right).$$



Recursive neural networks are different from recurrent neural networks:

- recursive neural networks are structurally recursive
- recurrent neural networks are cyclic in time.

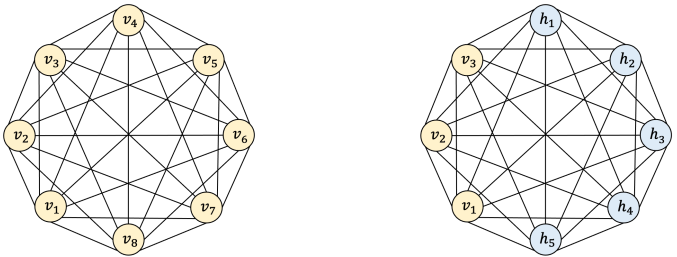
Recurrent vs. Recursive Neural Networks

	Recurrent neural networks	*Recursive* neural networks
Special vs. general	A special case of recursive neural networks	A generalization of recurrent neural network
Structure	Chain-like neural network structure	Tree-like neural network structure
Features	Recurring over time	Structurally recursive
Applications	Nature language processing, computer vision, and speech recognition	Nature language processing

Recurrent neural networks include RNNs, LSTM networks, and GRU networks.

Symmetric Neural Networks

In symmetric Neural Networks, the connections between units are symmetrical with the same weight in both directions.



Two types of symmetric neural networks: (a) Hopfield networks (without hidden units), (b) Boltzmann machines (with hidden units).

5 Connectionist Framework

Optimizations of Neural Networks

Multilayer artificial neural networks are composed of many artificial neurons connected in a certain way and layer, each with a weight between two layers of neurons.

The weight can be adjusted in some way, and then get the corresponding output value through the activation function, so that the neural network can adapt to the input and learn.

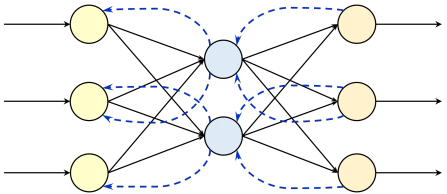
This is known as the *optimization* of neural networks, and is also often referred to as the *training* of neural networks.

Backpropagation

Backpropagation is short for “backward propagation of errors”.

Be a common method for training multilayer neural networks and adjusting network weight errors.

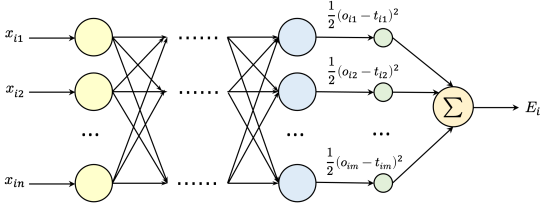
It usually combines with the gradient descent optimization method.



The blue dashed line is the path of backpropagation in the neural network.

Backpropagation

Consider a neural network with n dimensional input and m dimensional output.



Given $\{(\boldsymbol{x}_1, \boldsymbol{t}_1), \ldots, (\boldsymbol{x}_p, \boldsymbol{t}_p)\}$, minimize the error of actual $\boldsymbol{o}_i$ and target output $\boldsymbol{t}_i$,

$$E = \frac{1}{2} \sum_{i=1}^{p} \left\| \boldsymbol{o}_i - \boldsymbol{t}_i \right\|^2 = \sum_{i=1}^{p} \left\{ \frac{1}{2} \sum_{j=1}^{m} \left(\boldsymbol{o}_{ij} - \boldsymbol{t}_{ij} \right)^2 \right\}.$$

Backpropagation

E is calculated by combination of expanding network node function, a continuous differentiable function of the l weights $W_1, W_2, \ldots, W_l$.

It is possible to minimize E by using iterative process of gradient descent.

Each weight is updated using the increment ΔW_i.

$$\nabla E = \left\{ \frac{\partial E}{\partial W_1}, \frac{\partial E}{\partial W_2}, \cdots, \frac{\partial E}{\partial W_l} \right\}, \qquad \Delta W_i = -\gamma \frac{\partial E}{\partial W_i} \quad \text{for } i = 1, \ldots, l,$$

where γ represents the learning constant, which is a proportionality parameter. It defines the step length in the negative gradient direction for each iteration.

Evolutionary Methods

Another approach to neural network optimization, mainly including:

Evolutionary Algorithms

- Be population-based general metaheuristic optimization algorithms, inspired by biological evolution mechanisms.
- Typical evolutionary algorithms include genetic algorithms, genetic programming, evolutionary programming, gene expression programming, differential evolution, and evolution strategy.

Neuroevolution

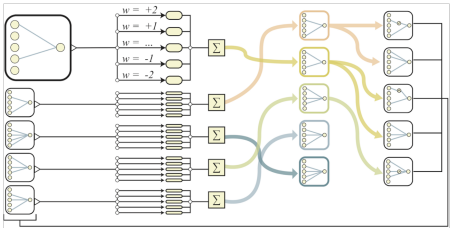
- It combines evolutionary algorithms with ANNs, i.e., using evolutionary algorithms to generate ANNs, including their parameters, topology, etc.

Weight-Agnostic Method

"Weight agnostic" refers to such ANNs with strong inductive biases, which can perform various tasks with randomly given initial weights, without training.

The algorithm is called neural architecture search (NAS):

- *Initialize*: creating initial population of minimal network topologies.
- *Evaluate*: evaluating each network over multiple rollouts.
- *Rank*: ranking networks according to performance and complexity.
- *Vary*: creating a new population by varying network topologies.



https://arxiv.org/abs/1906.04358

5 Connectionist Framework

Single- and Multi-Agent Networks

Single-agent networks

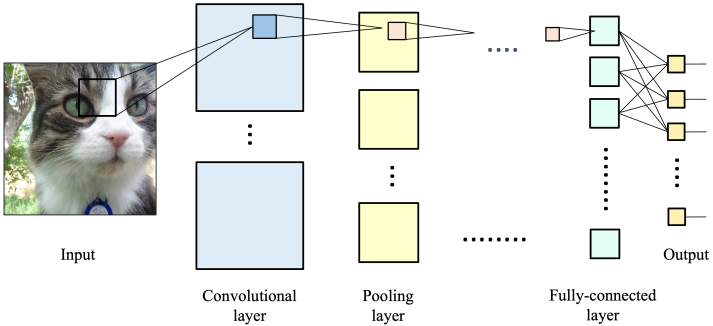
- Referring to the network that has the logical function of only one agent.
- Include convolutional neural network (CNN), LSTM network, GRU network, deep Boltzmann machine (DBM), and residual networks (ResNet).

Multi-agent networks

- Referring to the networks that have the logical functions of at least two agents, with the logical relationships between those agents.
- Include autoencoder (AE), generative adversarial network (GAN), collaborative neural network, and knowledge distillation network.

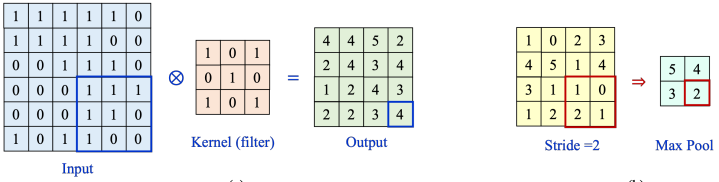
Case Study of Single-Agent Networks

CNN is a deep network, in addition to input and output layer, it includes several convolutional layers, pooling layers, and fully connected layers.



CNN four strategies: (i) local connection, (ii) shared weights, (iii) pooling, and (iv) multilayer.

Convolution and Pooling in CNN



(a) Convolution

An example for 6×6 image, and convolutional kernel (filter) size 3×3.

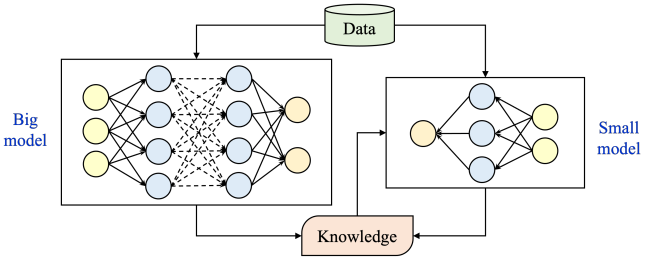
(b) Pooling

An example for max pooling, with stride 2.

Case Study of Multi-Agent Networks

In knowledge distillation network (KDN), there are two agents: large (teacher) model and small (student) model.

It uses the knowledge distillation method to transfer the knowledge from pre-trained large model to the small model.



Thank You