Japanese Journal of Radiology (2020) 38:907–921
https://doi.org/10.1007/s11604-020-00998-2

INVITED REVIEW

Introduction to deep learning: minimum essence required to launch a research

Tomohiro Wataya[1] · Katsuyuki Nakanishi[1] · Yuki Suzuki[2] · Shoji Kido[2] · Noriyuki Tomiyama[3]

Received: 11 March 2020 / Accepted: 2 June 2020 / Published online: 15 June 2020

Abstract

In the present article, we provide an overview on the basics of deep learning in terms of technical aspects and steps required to launch a deep learning research. Deep learning is a branch of artificial intelligence, which has been attracting interest in many domains. The essence of deep learning can be compared to teaching an elementaryschool student how to differentiate magnetic resonance images, and we first explain the concept using this analogy.Deep learning models are composed of many layers including input, hidden, and output ones. Convolutional neural networks are suitable for image processing as convolutional and pooling layers allow successfully performing extraction of image features. The process of conducting a research work with deep learning can be divided into the nine following steps: computer preparation, software installation, specifying the function, data collection, data edits, dataset creation, programming, program execution, and verification of results. Concerning widespread expectations, deep learning cannot be applied to solve tasks other than those set in specification; moreover, it requires a large amount of data to train and has difficulties with recognizing unknown concepts. Deep learning cannot be considered as a universal tool, and researchers should have thorough understanding of the features of this technique.

Keywords Deep learning · Convolutional neural network (CNN) · Artificial intelligence (AI) · Machine learning (ML) · Representation learning (RL)

Introduction

Artificial intelligence (AI) has been attracting greater attention in recent years. Successful research works are reported constantly in a variety of complex tasks, including image classification, object detection, speech recognition, playing games [1], and drawing pictures [2]. Related research in the medical field has been also progressing, including such research questions as analyzing electrocardiograms [3], detecting polyps from colonoscopy [4], and extracting features from pathological images [5].

Radiological image analysis and interpretation are the fundamental cognitive tasks in the field of radiology, which historically have been difficult to resolve despite technical advances in computer vision [6]. However, owing to the advancement of deep learning and other AI techniques and acceleration of computation using graphical processing units (GPUs) [7], the number of reports on radiological image analysis has increased dramatically. It is possible to mention such examples as suppressing bone shadows from chest radiography [8–10] and subtracting vessels from chest computed tomography (CT) images [11, 12]; several products have already obtained U.S. Food and Drug Administration (FDA) approval. In Japan, a product for detecting aneurysms based on magnetic resonance (MR) angiography was approved in 2019 by the Pharmaceuticals and Medical Devices Agency (PMDA) as the first application using deep learning [13, 14].

With the advancement of the AI technology, it becomes difficult even for radiologists to understand the technologies behind AI. People may express excessive expectations

✉ Tomohiro Wataya
wataya-osk@umin.ac.jp

[1] Department of Diagnostic and Interventional Radiology, Osaka International Cancer Institute, 3-1-69 Otemae, Chuo-ku, Osaka 541-8567, Japan

[2] Department of Artificial Intelligence Diagnostic Radiology, Osaka University Graduate School of Medicine, 2-2 Yamadaoka, Suita, Osaka 565-0871, Japan

[3] Department of Radiology, Osaka University Graduate School of Medicine, 2-2 Yamadaoka, Suita, Osaka 565-0871, Japan

concerning this technology, while others are worried to lose their jobs due to this reason. These issues can be attributed to the fact that there are a limited number of articles that briefly summarize the key concepts of AI, including technical aspects.

In the present article, we review the basics of AI, specifically, focusing on the deep learning technology by defining the key terms. We present our own intuitive explanation of deep leaning and describe its basic concepts and discuss the background for convolutional neural networks (CNNs). We also suggest the concrete steps to launch a research work dedicated to deep learning. In addition, we describe the technical limitations associated with deep learning.

Definitions

In this section, we define the key terms of computer science. This will be helpful to understand the relationship between AI and deep learning.

Artificial intelligence (AI)

AI is a class of computer software capable of performing tasks that require imitating human intelligence [6]. Although the range of tasks corresponding to the category of AI is rather comprehensive, it is often regarded to relatively complex tasks.

Machine learning (ML)

ML is a subfield of AI in which algorithms are trained to perform tasks by learning patterns from the data in question rather than by applying explicit programming [7].

In the conventional AI techniques, human experts statistically or empirically extracted features from the considered data and encoded them into algorithms. The AI software is capable of processing inputs using a manually determined scale and formulating required answers. Meanwhile, in ML, it is necessary to program the methods to automatically analyze the inputs and derive the answers instead of formulating manual-based criteria.

Technical examples of ML include k-nearest neighbor [15], support vector machine (SVM) [16], decision tree [17], and the Naive Bayes algorithm [18].

Representation learning (RL)

RL is a subfield of ML. ML depends greatly on preprocessing pipelines and data transformations which result in a representation of the data, whereas RL does not [19]. However, the RL software can learn representations of the data on its own, which allows gradual improvement of the quality and accuracy of the outputs through the training process. An artificial neural network (ANN) [20] is a typical example of RL.

Artificial neural networks (ANNs), deep learning, and convolutional neural networks (CNNs)

ANNs were inspired by the concept of human neural networks. Virtual neurons line up to compose a unit denoted as a layer, and combinations of layers and connections between them constitute a network model. When the model employs many layers, typically more than 20 [7], the technique is called deep neural network (DNN) learning, or deep learning. CNN is a branch of neural networks that employ the convolution technique. It is commonly applied to computer vision tasks.

An intuitive explanation of deep leaning and its main concepts

To understand deep learning intuitively, let us consider a friendly analogy.

Let us imagine that we need to teach an elementary school student how to differentiate the T1-weighted image (T1WI) and T2-weighted image (T2WI) slices of brain magnetic resonance imaging (MRI). As the student is not aware of the anatomy of the brain or the MRI technology in detail, the explanations using anatomical or technical terms, such as "Water is black on T1WI and white on T2WI", would not work, and therefore, it is necessary to change the strategy.

A new strategy is represented in Fig. 1a. We put two boxes in front of the student. For convenience, we denote them box 0 and 1, respectively. We expect the student to put the T1WI slices into the box 0 and T2WI into the box 1. Then, we pass one slice (step 1), noting "Put this image into either of the boxes". At first, the student does not understand what he or she is being asked for, so he or she randomly selects a box and puts the image into it (step 2). After this selection, we check the provided answer (step 3) and tell if the selection has been correct or not (step 4). By repeating such trials over and over again, the student gradually learns what box 0 and 1 images correspond to. However, it does not mean that the student understands the physical definition of the T1WI and T2WI or can specify the difference between the T1WI and T2WI slices. This means that to evaluate the validity and usefulness of the learned concept, the student needs to undergo examinations using new image data. In exams, we check the provided answer after the student's selection, but we do not expose the ideal answer (step 4 is omitted). If there is a large difference in the accuracy between the study ("training") process and the exam ("validation") process, the student's understanding is considered as non-adjustable to new unseen images.

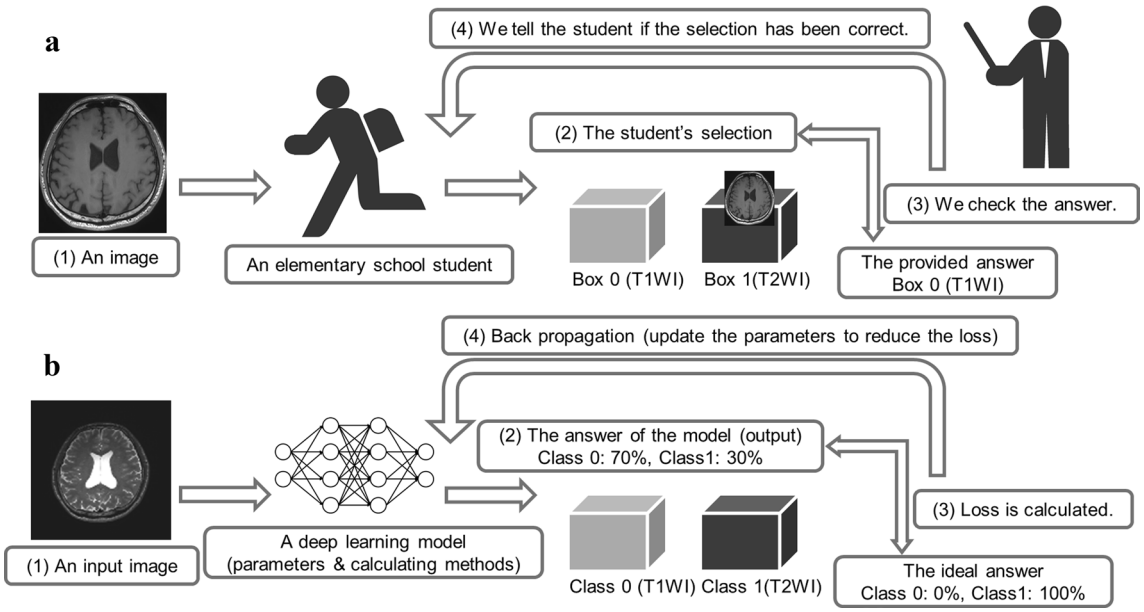

Fig. 1 Steps to teach **a** an elementary school student or **b** a deep learning model how to differentiate the T1WI and T2WI slices. The numbers in the figure correspond to the step numbers in the text

Almost the same is true for deep learning (Fig. 1b). Deep learning models are composed of calculating methods and millions of or, sometimes, billions of parameters, mimicking human brain neural networks. When we input an image into a model (step 1), it generates its answer (step 2). However, the output is not precisely the box number, but the possibility of each box ("class"). The class with the highest possibility is considered as the answer of the model. An appropriate model is required to provide accurate and clear-cut answers. Therefore, we introduce a "loss function" to quantify the extent of difference between the model output and the ideal answer denoted as "loss" (step 3). In the training process, the model parameters are updated to reduce the loss (step 4). This update process is referred to as "back propagation". The model is trained based on the same data many times, and the repeated time is called "epoch". Then, in the validation process, the loss is calculated, but we do not update the model parameters (step 4 is omitted). The ideal result is that in both training and validation trials, the accuracy would increase monotonically, and the loss would decrease. When the model is overfitted with respect to the training data and cannot be adjusted to the validation data, the difference of the loss between the training and the validation gradually expands, which is denoted as "overfitting".

Main tasks for deep learning and well-known models

In this section, we introduce several deep learning tasks related to image processing along with the well-known models. The important aspect to be noted here is not how well the models can perform in the task, but whether the task inputs and outputs are suitable for deep learning.

Classification

In the classification task, the model receives an image and predicts what is depicted in the image. For example, goals for the classification tasks can be as follows: evaluating a digit in a handwritten single digit image [21], classifying diffuse lung disease patterns [22], and estimating MRI sequences [23].

Many deep learning models, specifically, CNN models, have been proposed to resolve this task. Examples of the well-known models are LeNet [21], AlexNet [24], visual geometry group net (VGGNet) [25], and residual neural network (ResNet) [26].

Regression

In the regression task, the model predicts one value from an image. For example, studies have been conducted on estimating the bone age from radiographs of fingers [27], and estimating the risk of cardiovascular disease from fundus images [28].

In the regression task, network models used in the classification task can be adjusted by replacing the output data format (output layer).

Object detection

In the classification task described above, the classification model is not applicable when more than one objects exist in one image and cannot specify the location of each object in the image. In the object detection task, the positions of objects are represented by rectangles (bounding boxes) circumscribing the objects in the image. An example of detecting ventricles from a T2WI slice (Fig. 2a) is shown in Fig. 2b.

Various networks, such as regions with convolutional neural networks (R-CNN) [29], fast R-CNN [30], single shot multibox detector (SSD) [31], and you only look once (YOLO) [32], have been proposed. These algorithms often include the two major steps: extracting a large number of rectangular candidate areas from the image, and classifying what kind of objects which they are [33].

Segmentation

In the object detection task, locations of objects can be estimated; however, the shapes of the objects are not known. The purpose of the segmentation task is to extract the areas having the target structure presented in the image, if any. Figure 2c provides an example of segmentation of ventricles from a T2WI slice. Such networks as fully convolutional networks (FCN) [34], SegNet [35], and U-Net [36] have been proposed to solve this problem.

Super-resolution

In this task, high-resolution images are obtained from low-resolution images. Object contours in the magnified images are blurred with simple linear interpolation, whereas super resolution keeps the sharpness of edges [37]. CNN models, such as super-resolution using deep convolutional networks (SRCNN) [38], can be used for this purpose. In medical imaging, this technique is used for various modalities, including chest CT [39] and mammogram [40].

Generating images

In recent years, generative adversarial networks (GANs) have attracted great attention as an approach to generate images. In GANs, a generator, which produces fake images from randomly created noise, and a discriminator, which is used to distinguish fake images from real one, are alternately trained, so that the generator can produce a larger number of close to real images [41]. Various GAN-related methods have been proposed [42], such as conditional GAN (cGAN) [43] and cycleGAN [44].

Extracting features

Autoencoder is considered as a useful approach for extracting features from images denoted as "dimensionality reduction". This network is composed of encoder and decoder. The encoder gradually compresses images and finally loads them into a layer denoted as "code layer", which is composed of the smallest number of neurons. In its turn, the decoder expands the data in the code layer, restoring the input image. When both networks are trained successfully, the data in the code layer can be used to represent image features [45].

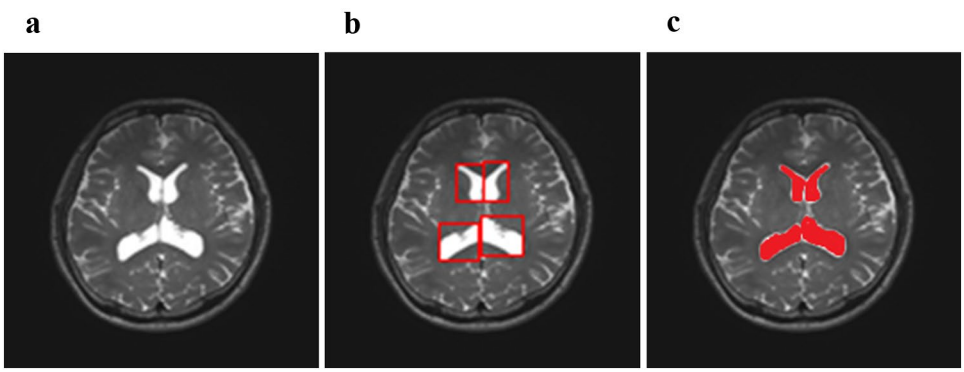

Fig. 2 Difference of outputs in object detection and segmentation tasks. In the example task of detecting ventricles from **a** a T2WI slice, **b** object detection task networks are represented by bounding boxes and **c** segmentation task networks correspond to ventricles. The outputs are manually reproduced by the author

Fig. 3 Pixel values in a monochrome image. Images are composed of pixels, and each pixel has its own value. The image is a handwritten number from MNIST database [46]

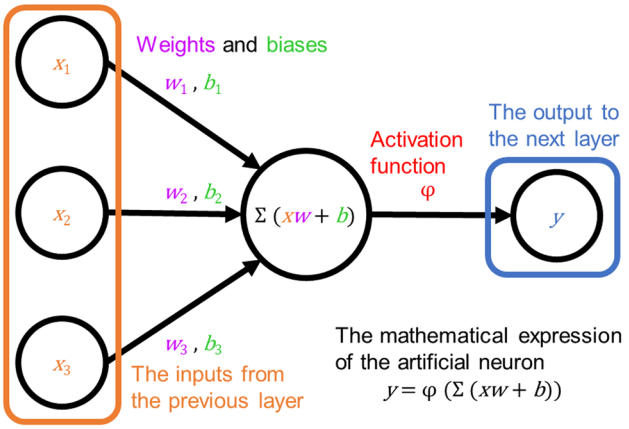

Fig. 4 Process in an artificial neuron in the affine layer. An artificial neuron receives the data from the previous layer and outputs a value according to the parameters (weights and biases)

Components of deep learning models

Deep learning models are composed of many layers. The first one is denoted as the "input layer"; the last one is referred to as the "output layer"; and the others in the middle are "hidden layer(s)". In this section, we explain the function of each layer in detail.

Input layer

This layer is the first layer of the network, which receives the input data. When images are employed as the input data, the input data are the matrices of values (Fig. 3). When color images are used, the image is considered as the complex of the three channels (red, green, and blue) and the input data triple. Each artificial neuron in this layer receives only one value from the input. This means that the input layer needs the same number of neurons as the number of values.

Hidden layer

There are many kinds of hidden layers. In this section, we introduce several common ones.

Affine (dense) layer

In the affine layer, artificial neurons are lined up in a row, so we consider the concept of artificial neurons in more details (Fig. 4).

Each neuron receives the input data from all neurons in the previous layer. For each input from the previous layer (outputs of the previous layer) x, the neuron has a weight w and a bias b, and "$xw+b$" is calculated and summed up. An "activation function" is used to obtain the output y from the sum. The output is used as an input in the next layer. Activation functions are the non-linear functions. Examples of the well-known activation functions are sigmoid (Fig. 5a), tanh (Fig. 5b), and rectified linear unit (ReLU) [47] (Fig. 5c). In this layer, weights and biases are the trainable parameters and are updated in the training phase.

Due to the process described above, the affine layer processes the images just as an assembly of the scattered data, namely, the output does not consider the location datum of each pixel nor combinations among a neighborhood of points [48]. Moreover, even when an object in an image moves by a single pixel, since all processing neurons shift, the output can vary significantly.

Convolutional layer

Convolution has been considered in the conventional imaging technology before the development of deep leaning, such as to blur images or to extract contours (Fig. 6).

In convolutional layers, small grids of weights referred to as "kernels" are applied to process the input. The kernels refer to the same shape of an area in the inputs, constituting the sum of the pixelwise multiplications of the inputs and the weights.

An example of applying a 3×3 kernel to a 5×5 input is provided in Fig. 7. First, we apply the kernel to the left top of the input. The sum of the nine multiplications (as shown in the figure) is written on the left top of the output (Fig. 7b). Then, we move the kernel to the right (in this case by one pixel) to calculate the next sum (Fig. 7c). By repeating this process, we fill in the output matrix (Fig. 7d). Although the output becomes smaller than the input in this case, there are the methods proposed to keep the data size, such as filling the peripheral data by zero.

In the actual convolutional layer, more than one kernels are often used, so that the number of output matrices is the same as that of the kernels. In this layer, trainable parameters denote the weights of the kernels and, if any, biases.

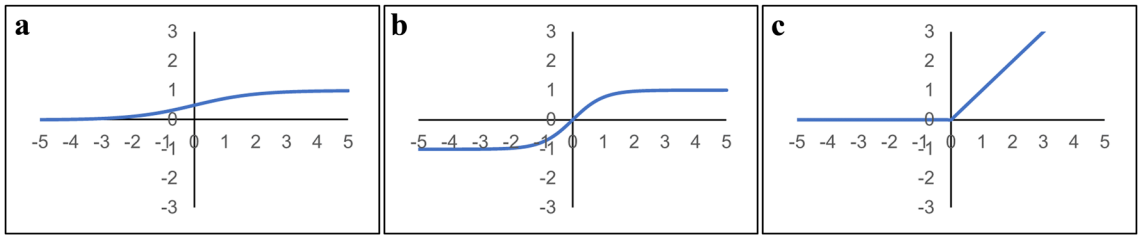

Fig. 5 Typical activation functions in neural networks. Each graph shows the shape of **a** sigmoid, **b** tanh, and **c** ReLU functions, respectively

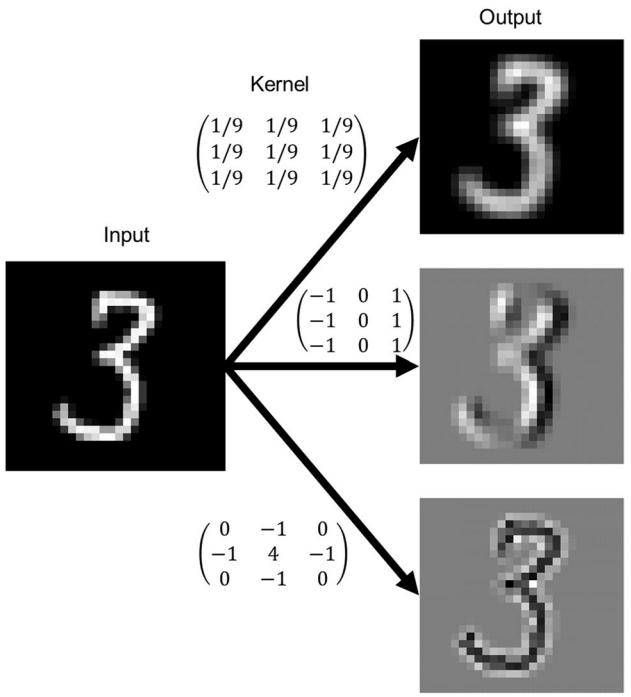

Fig. 6 Examples of the convolution process in the conventional imaging technology. The output varies widely depending on the kernel, and various kernels have been developed. The input image is a handwritten number from MNIST database [46]. Processed with a program proposed in [49] (kernel parameters are manually changed by the author)

An activation function, commonly the ReLU function, is similar to affine layer.

By introducing the convolutional layers, the models are considered able to extract local features of the image.

Pooling layer

In the pooling layer, a process of reducing the image size, while maintaining the characteristics of the image, is performed. An example of a process in a pooling layer for the case of focusing on a 2 × 2 area is presented in Fig. 8. When we focus on the left top of the input area, the maximum value is written to the left top of the output data (Fig. 8b).

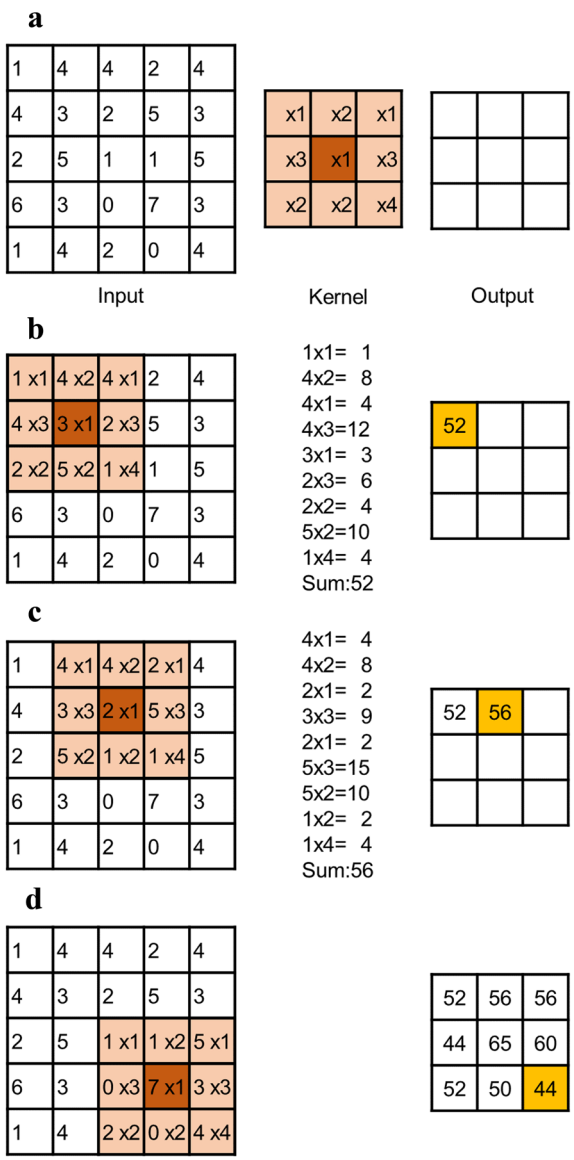

Fig. 7 An example of the convolution process. **a** Input data, a kernel (weights), and an output matrix (currently empty). **b** The kernel is adapted to the left top of the input, and the left top of the output is filled. **c** The kernel is moved, filling another part of the output. **d** By repeating these steps, the output matrix is fully filled

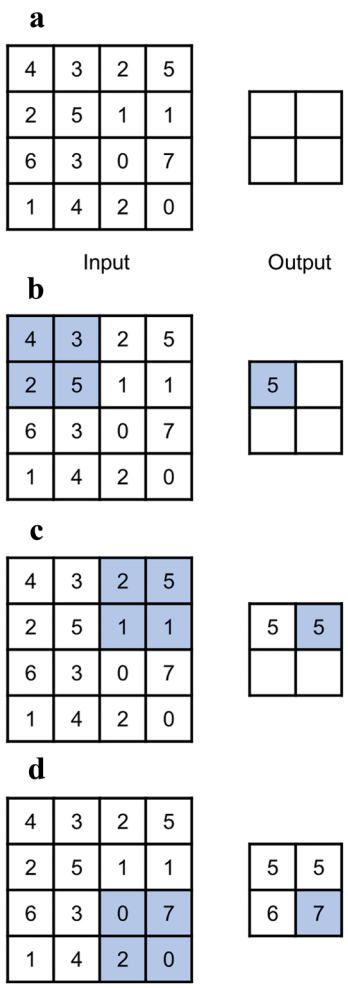

Fig. 8 An example of the max pooling process. **a** Input data and an output matrix (currently empty). **b** We focus on the 2×2 area of the left top of the input and fill the left top of the output by the largest value. **c** We move two pixels to the right, filling another part of the output. **d** By repeating these steps, the output matrix is fully filled

Next, we move two pixels to the right and record the maximum value (Fig. 8c). This is repeated to fill all output data (Fig. 8d). Although this example corresponds to the "max pooling layer" that transmits the maximum value of the focusing area to the next layer, there are other types of layers to consider, such as an "average pooling layer" that transmits the average.

There are no trainable parameters defined in this layer. The output is considered to represent the condensed features of the image. Furthermore, the whole network can be less sensitive to the slight differences in locations of objects in the image [48] and robust against data noise, if any.

In actual CNNs, convolutional and pooling layers can be used alternately, so that the feature extraction and condensation are performed.

Output layer

This is the final part of the network. In this layer, the calculation results of the previous hidden layer are combined into the final network output. This layer is often defined as the affine or the convolutional layer; however, the number of neurons and the suitable activation function are determined according to the task. Common structures of the output layer for particular tasks are listed in Table 1.

Loss function

Strictly speaking, the loss function is not included in the neural network. It quantifies the difference between the model output and the perfect answer. Well-known loss functions include binary cross entropy, categorical cross entropy, and mean squared loss. Similar as in the output layer, the selection of a suitable loss function is limited depending on the particular task (Table 1). For complex tasks, it may be necessary to define own loss function.

Optimizer

The optimizer is an algorithm employed in a training process to find the optimal combination of the trainable parameters that minimizes the loss. Figure 9 shows an example of the stochastic gradient descent (SGD) [51]. The curve in the figure is a pattern diagram of the relationship between the parameter w and loss L when an image is input to the model. Let us assume w is w_0. The gradient g_0 of the L at the point w_0 is calculated using the differential of w. Herein, we introduce a constant denoted as "learning rate" η and "w_0—ηg_0" is the revised parameter w_1. The

Table 1 Examples of the output layers and loss functions for common tasks

Task	Classification task		Regression task
	With two classes	With more than two classes	
Type of the layer	Affine layer	Affine layer	Affine layer
Number of neurons	One	The same as the number of the classes	One
Activation function	Sigmoid function	Softmax function	No activation function
Loss function	Binary cross entropy	Categorical cross entropy	Mean squared loss

This table is created based on description in [50]

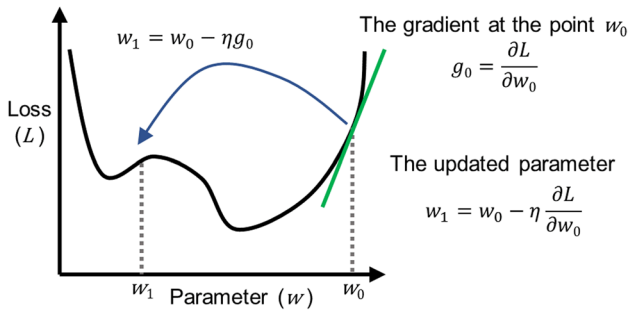

Fig. 9 A pattern diagram of updating parameter. The parameter w is updated based on the differential of the loss L. The symbols correspond to the text

shape of the curve varies according to the image. Therefore, we gradually modify w by exchanging the input.

The process includes the differential of a composite function and this optimization is performed following the inverse order starting from the output layer to input layer. Therefore, the optimization is called backpropagation. The SGD and several other algorithms have been proposed, such as momentum [51], RMSProp [52], AdaGrad [53], and Adam [54].

We demonstrated a method to update the parameters per image. However, "mini-batch training" is a commonly used method. In this technique, some (usually from 10 to 100) images are randomly selected and the gradients for each input are averaged. This is efficient, because the loss over a mini-batch is an estimate of the gradient over the entire training data [55].

An example of CNN

An example of CNN definition in Keras [56] is represented in Fig. 10a. A network diagram for the same network is provided in Fig. 10b. This example is provided for a classification task corresponding to processing 32 × 32 px monochrome images into ten classes and is based on LeNet [21] (partially modified by the author). Here, the input layer receives 32 × 32 data pixel by pixel. In this case, the explicit definition of the input layer is omitted. Instead, the shape of the input data is specified in the first layer. Thereafter, the data pass through the convolutional layer (Conv2D) and max pooling one (MaxPooling2D) alternately. Furthermore, the data pass through several affine layers (Dense) and then are aggregated gradually into ten neurons in the output layer. In this example, the activation functions are defined as ReLU in the convolutional layers and tanh in the affine layer. As

a

```
model = Sequential()
model.add(Conv2D(filters=6, kernel_size=5, input_shape=(32,32,1), activation="relu"))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(filters=16, kernel_size=5, activation="relu"))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(units=120, activation="tanh"))
model.add(Dense(units=84, activation="tanh"))
model.add(Dense(units=10, activation="softmax"))
model.compile(loss="categorical_crossentropy", optimizer=Adam(), metrics=["accuracy"])
```

b

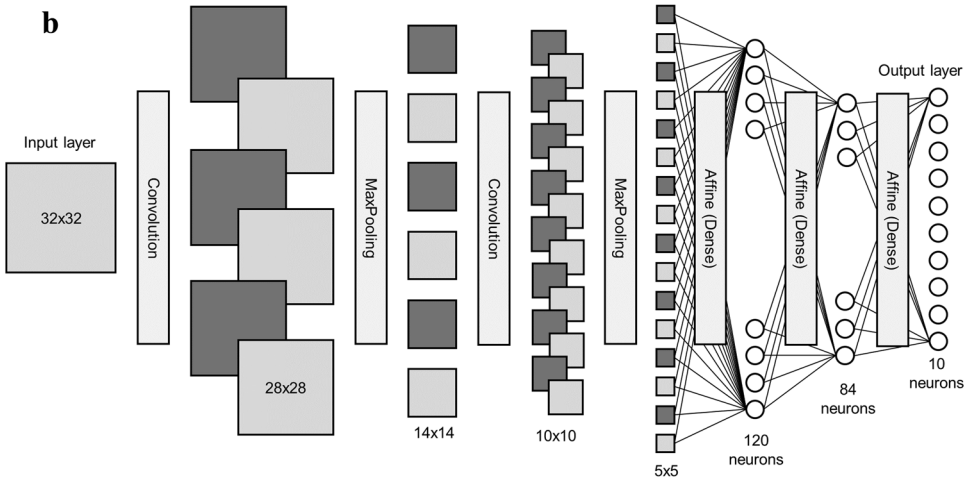

Fig. 10 An example of LeNet definition. **a** A program example is implemented in Keras. The convolutional layer convolution (Conv2D) and max pooling (MaxPooling2D) layers are used alternately in the first half of the model. The latter half of the model is composed of affine (Dense) layers. Finally, a loss function and an optimizer are defined. **b** A network diagram of the network defined in **a**. Some of the neurons and data connections are omitted in the figure

this is a classification task with more than two classes, the softmax function is used in the output layer and categorical cross entropy is used as the loss function. In this case, we selected Adam as the optimizer.

Steps to launch research with deep learning

In this section, we explain the specific steps which we propose for the deep learning research works.

Computer preparation

In theory, most computers are suitable for the deep learning research regardless of the operating system (OS) and the computational power. However, the computers with at least one GPU (Fig. 11) are preferable. Central processing units (CPUs), which are presented in all computers, are available for general purposes. However, although GPUs can solve only simple calculations, they can do much faster than CPUs, and this feature is adequate for deep learning researches. GPUs are not always necessary, but they enable fast experiments and studies.

However, we must understand that some GPUs, especially old types, are unavailable for the studies. In addition, particular GPUs are not compatible with some OSes as the required GPU drivers are not provided.

It should be noted that various online cloud systems, such as Google Colaboratory [58], Microsoft Azure [59], and Neural Network Console [60] (listed in an alphabetical order), are available. These systems provide affordable workspaces with GPUs and installed software; therefore, depending on budget and research objectives, these systems can be considered as suitable options.

Fig. 11 A photograph of a graphic processing unit (GPU). The image is reprinted from [57] with permission by NVIDIA

Software installation

An example of a necessary software list formulated when using TensorFlow [61] and Keras [56] on Windows is provided in Table 2. The list may be different depending on the characteristics of GPUs, and the GPU maker is supposed to be NVIDIA in this example.

More than one versions are usually available for some software exemplars; however, due to the compatibility issue between various software programs and versions, the combination of the most recent versions may not be applicable. It is necessary to review the information on the homepages of software providers.

Specifying the function

In this step, it is required to clarify the problem to be solved. This step designates the format of the input and output data. In addition, specifically, when the research is focused on image processing, it is necessary to define the image size, monochrome or color image, and the image format, such as portable network graphics (PNG), tag image file format (TIFF), or digital imaging and communications in medicine (DICOM).

Data collection

Similarly as in other medical research works, it is necessary to search target cases and collect data. When required, it is possible to create the answer (teacher) data.

Data edits

It is necessary to edit the collected data according to the provided specifications. For example, according to the specification, DICOM format images have to be converted to PNG images, or color images have to be converted to monochrome images. To unify the image size, the image may be enlarged/reduced or trimmed as appropriate.

Table 2 Example of the software installation list

Without GPUs	With GPUs
	CUDA
	cuDNN
	Visual Studio
Anaconda	Anaconda
TensorFlow	TensorFlow (GPU version)

Jupyter Notebook, keras, HDF5 for python, matplotlib, Pillow, pandas, SciPy, Spyder, scikit-learn, Cython, OpenCV, pydicom

The list is created based on description in [77]. References are [56, 62–76]. Some software is not always necessary

Dataset creation

It is required to link the input and output of the collected data. For example, in a classification task, it is possible create a comma-separated values (CSV) file to define combinations of a file name and a correct class; otherwise, it is possible to divide the images into folders according to the correct classes.

As described above, in addition to the training data used for updating the model parameters, the validation data are required to detect overfitting. Although it is possible to divide them randomly at the program execution stage, the training and validation datasets can be created separately. Among the collected data, 80% is often assigned to the training data and 20% to the validation; however, this value is not precisely defined and can be selected as appropriate according to the purpose of the study.

Programming

To implement deep learning, it is necessary to create a corresponding program. It seems rather difficult for people who are not used to program professionally; however, examples of programs corresponding to the common tasks are available in the related technical literature or in the Internet. It is possible to copy and paste the examples provided in open code and change several particular parameters (image size, the number of classes, etc.). In such way, it is possible to create simple programs.

In recent years, environments providing graphical user interface (GUI), such as DIGITS [78] (Fig. 12) and Neural Network Console [60] (Fig. 13) (listed in an alphabetical order), have been developed, and therefore, programming is not necessarily essential.

Program execution

To obtain required results, it is necessary to execute the implemented program and train the model. The required computational time may vary greatly depending on the contents of a task, the image size, and the performance of GPUs. A learning curve can be monitored (Fig. 14). When the training is performed successfully, the accuracy of both training and validation gradually improves, and the corresponding loss decreases (Fig. 14a). However, when the gap between training and validation expands, this effect is referred to as overfitting (Fig. 14b). The learning can be terminated when overfitting has occurred; or when the learning has converged, and the accuracy has not improved.

Verification of results

In this step, it is necessary to check whether the learning is successful. Herein, we present some technical clues to improve or verify the results.

Techniques for scarce medical images

Collecting a large number of medical images with accurate labels can be a daunting task. Therefore, we present some techniques that can be used to work with scarce data.

In the technique called "data augmentation" [79], image variants can be created by rotating, enlarging, or reducing the original data. The variants appear the same at first glance. However, they can make big differences in the AI process. Therefore, it is expected that a few original images allow effective learning.

Another technique is "transfer learning" [80]. We pretrain a model using a large number of non-medical images or formulate a model using a set of successful parameters of winners in the popular ML competitions, such as the ImageNet challenge. Such models are proficient at processing general images, and by fine-tuning them with the target medical data, they are expected to mark good grades.

Techniques for preventing overfitting

Overfitting occurs when there are not enough training data or when the model contains large number of parameters [51].

Since the weights tend to be larger in case of overfitting, forcibly keeping the values small, which is called "weight decay", was proposed as a key to overfitting. This is achieved by adding the L2 norm (square root of the sum of the squares) of the parameters to the loss. Thus, reducing the loss by backpropagation can result in small parameters, since the norm works as a penalty for big parameters [51].

Another technique is "dropout". During training, certain percentage ("dropout rate") of neurons are randomly selected for each mini-batch, and those neurons drop out of training. Through this process, the model becomes a virtually "thinned" network and can be robust against overfitting [81].

The final technique is "batch normalization". Batch normalization layers are often inserted to normalize the intra-mini-batch data those flow into the activation functions. Although this technique was first aimed to accelerate the training, it also works as a powerful aid for overfitting [55].

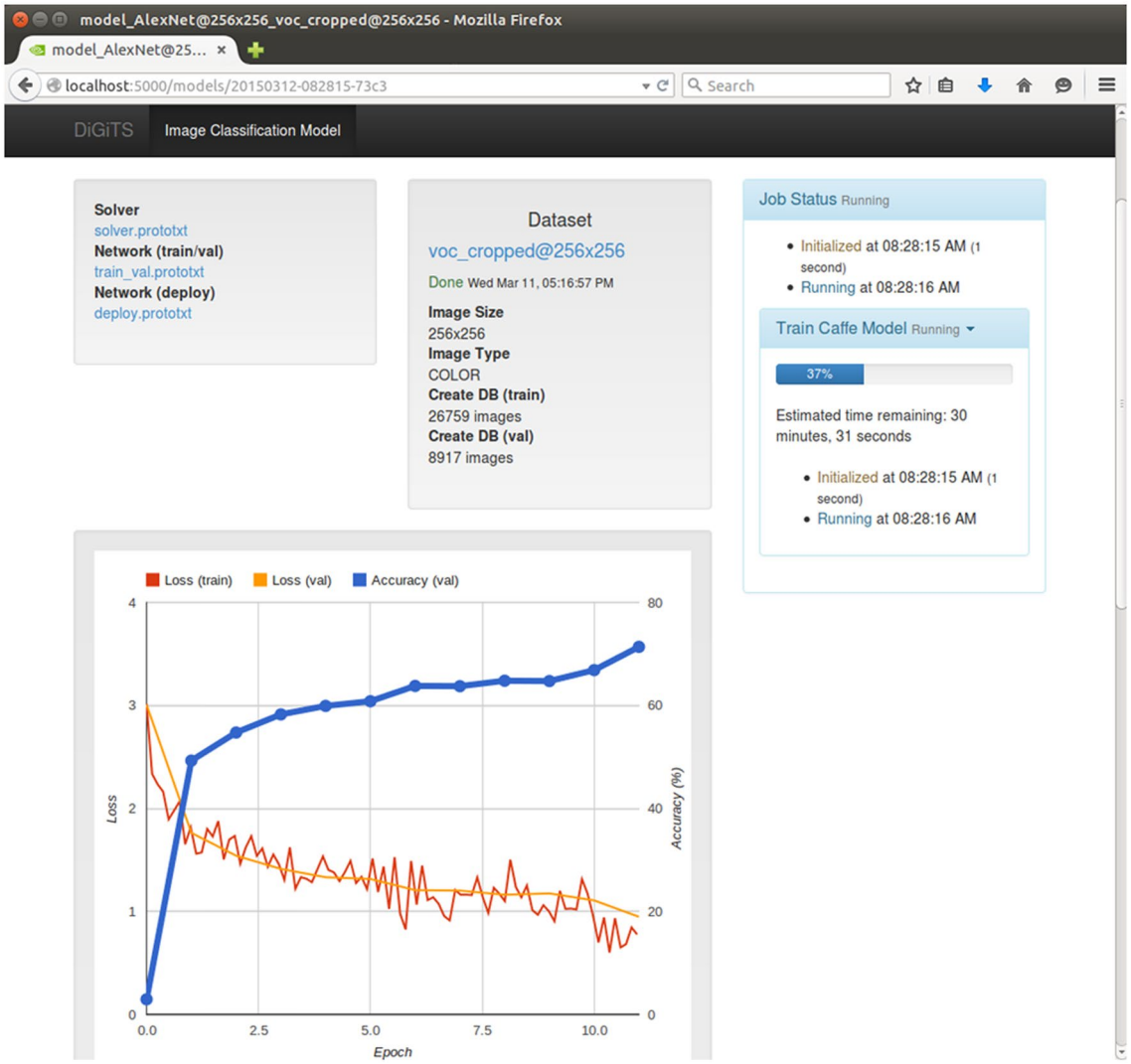

Fig. 12 A screenshot of DIGITS. An example of GUI application for deep learning. The image is reprinted from [57] with permission by NVIDIA

Tuning hyperparameters

The models contain millions of parameters. Some of these parameters are non-trainable and cannot be modified automatically. Such parameters are epoch, mini-batch size, learning rate, dropout rate, parameter initialization values, and selection of the loss function or optimizer. By tuning these hyperparameters, we can seek better sets of parameters even with the same models and data.

Visualization of reasons for judgement

The meanings of each parameter or process in the model are still unclear. However, gradient-weighted class activation mapping (Grad CAM) was proposed as a practical method [82]. Additionally, layer-wise relevance propagation (LRP) was proposed [83]. The aforementioned methods aimed to determine which image properties which the model was focused on. In particular cases, the model considered the trivial image artifacts (shading of the background or fonts and positions of characters appearing in the image, if any) to formulate an answer. It is necessary to determine the capability of the model in providing relevant answers, based on a reasonable rationale.

Limitations of deep learning

Although it is considered that AI, including deep learning, is expected to substitute several radiologists' work processes in the future, the technique itself has many limitations.

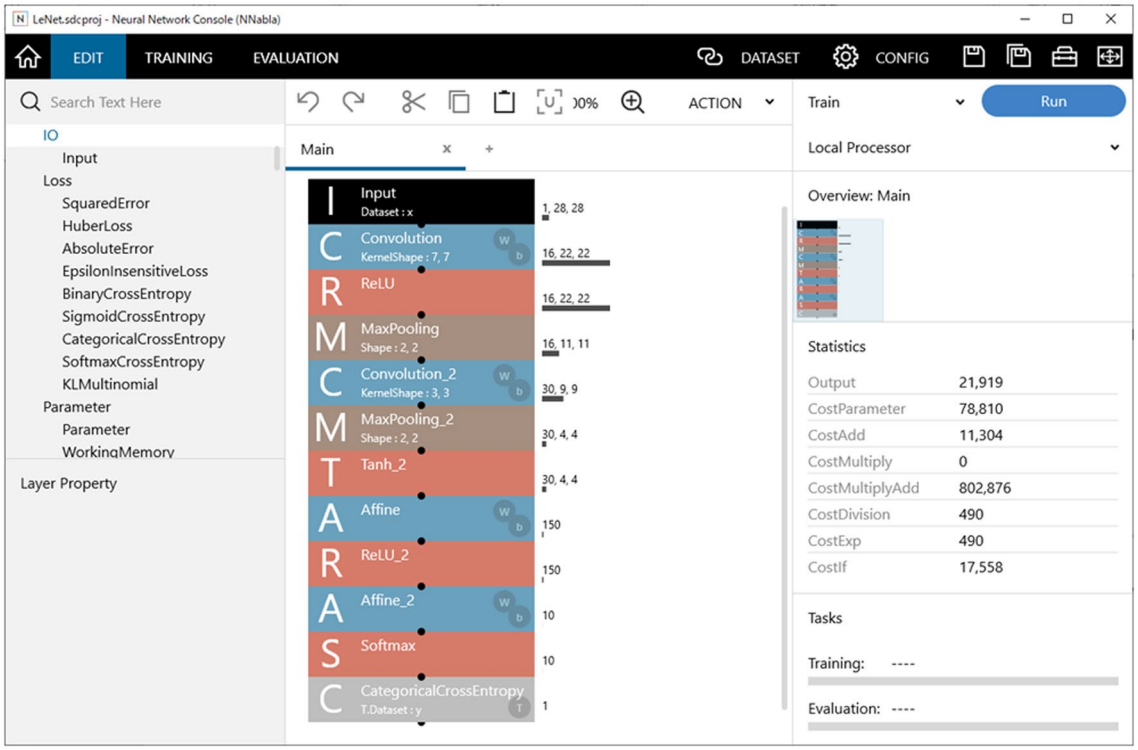

Fig. 13 A screenshot of Neural Network Console. An example of GUI application used for deep learning. The screenshot is created by the author with permission by Sony Network Communications

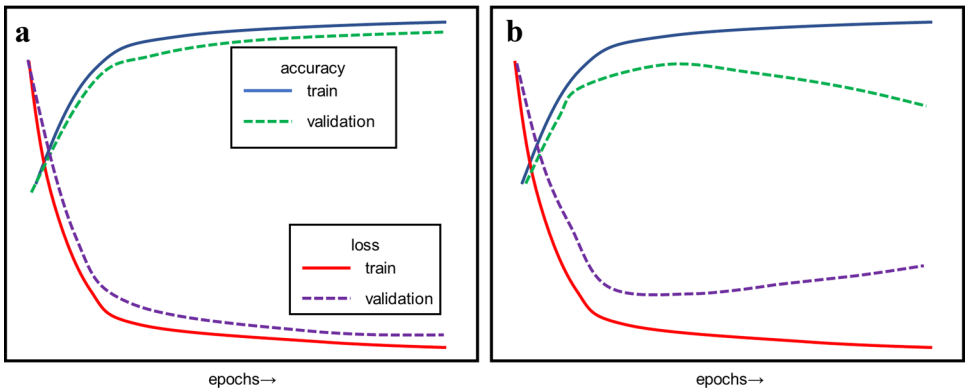

Fig. 14 Pattern diagrams of a learning curve. **a** A pattern of a successful study. There is little difference between the training and validation curves. **b** A pattern of overfitting. The difference between the curves gradually expands

First, it cannot solve tasks other than those set in specification. A network implemented for the classification task cannot be applied to the regression problem, and an image of the size other than defined in the specification cannot be input or output. Radiologists usually diagnose images based on the knowledge from various sources, such as scientific papers, books, their own experience, study sessions, etc. However, deep learning cannot be implemented based on all these resources, as the corresponding data formats are not unified and therefore, at present, deep learning is able to process only the data limited at some extent.

Second, the task of training networks requires a large amount of the relevant case data. It is often rather difficult to collect the exact data without bias between facilities and imaging devices. In addition, when a new disease concept appears, and a corresponding novel technique for the disease is being developed, data collection can become a serious barrier, specifically, for rare diseases.

Furthermore, in deep learning, there is a difficulty associated with recognition of unknown elements. The model forcibly processes the data as long as the input format matches the specification, even when the input data are not relevant

for the considered task, thereby providing meaningless answers.

Conclusion

In the present article, we reviewed the technical basics of deep learning using our own intuitive analogy, explained the structure of models succinctly and proposed the steps to initiate a deep learning research.

In recent years, many brilliant achievements in deep learning have been reported; however, they cannot be considered as universal tools. Considering that AI will become more prevalent in daily clinical practice in the near future, we note that medical doctors need to have clear understanding on the advantages and disadvantages of these approaches.

It should be noted that at present, in the radiology departments, most of the data used for daily work are digitized, and the applicability to information technology (IT) research studies is better compared with other departments. Radiologists can access and collect radiological images and can create the own teacher data; therefore, it is important to take the initiative to actively engage in AI-related research.

Acknowledgements The authors would like to thank Enago (www.enago.jp) for English language review. The authors would like to acknowledge NVIDIA for providing Figs. 11 and 12. The authors are grateful to Soichiro Tateishi, a radiological technologist in Osaka International Cancer Institute, for providing sample MR images in Figs. 1 and 2.

Compliance with ethical standards

Conflict of interest Yuki Suzuki and Shoji Kido receive research funding from Fujifilm Co., Ltd., but all the authors declare no conflicts of interest associated with this manuscript.

References

1. Silver D, Huang A, Maddison CJ, Guez A, Sifre L, van den Driessche G, et al. Mastering the game of Go with deep neural networks and tree search. Nature. 2016. https://doi.org/10.1038/nature16961.
2. The Next Rembrandt [Internet]. [place unknown]: The Next Rembrandt. https://www.nextrembrandt.com/. Accessed 8 Mar 2020.
3. Attia ZI, Noseworthy PA, Lopez-Jimenez F, Asirvatham SJ, Deshmukh AJ, Gersh BJ, et al. An artificial intelligence-enabled ECG algorithm for the identification of patients with atrial fibrillation during sinus rhythm: a retrospective analysis of outcome prediction. Lancet. 2019. https://doi.org/10.1016/S0140-6736(19)31721-0.
4. Mori Y, Kudo SE, Misawa M, Saito Y, Ikematsu H, Hotta K, et al. Real-time use of artificial intelligence in identification of diminutive polyps during colonoscopy: a prospective study. Ann Intern Med. 2018. https://doi.org/10.7326/m18-0249.
5. Yamamoto Y, Tsuzuki T, Akatsuka J, Ueki M, Morikawa H, Numata Y, et al. Automated acquisition of explainable knowledge from unannotated histopathology images. Nat Commun. 2019. https://doi.org/10.1038/s41467-019-13647-8.
6. Chartrand G, Cheng PM, Vorontsov E, Drozdzal M, Turcotte S, Pal CJ, et al. Deep learning: a primer for radiologists. Radiographics. 2017. https://doi.org/10.1148/rg.2017170077.
7. Erickson BJ, Korfiatis P, Akkus Z, Kline TL. Machine learning for medical imaging. Radiographics. 2017. https://doi.org/10.1148/rg.2017160130.
8. Schalekamp S, van Ginneken B, Karssemeijer N, Schaefer-Prokop CM. Chest radiography: new technological developments and their applications. Semin Respir Crit Care Med. 2014. https://doi.org/10.1055/s-0033-1363447.
9. Schalekamp S, Van Ginneken B, Koedam E, Snoeren MM, Tiehuis AM, Wittenberg R, et al. Computer-aided detection improves detection of pulmonary nodules in chest radiographs beyond the support by bone-suppressed images. Radiology. 2014. https://doi.org/10.1148/radiol.14131315.
10. ClearRead Xray [Internet]. Miamisburg: Riverain Technologies. https://www.riveraintech.com/clearread-xray/. Accessed 8 Mar 2020.
11. Lo SB, Freedman MT, Gillis LB, White CS, Mun SK. JOURNAL CLUB: computer-aided detection of lung nodules on CT with a computerized pulmonary vessel suppressed function. Am J Roentgenol. 2018. https://doi.org/10.2214/AJR.17.18718.
12. ClearRead CT [Internet]. Miamisburg: Riverain Technologies. https://www.riveraintech.com/clearread-ct/. Accessed 8 Mar 2020.
13. Ueda D, Yamamoto A, Nishimori M, Shimono T, Doishita S, Shimazaki A, et al. Deep learning for MR angiography: automated detection of cerebral aneurysms. Radiology. 2019. https://doi.org/10.1148/radiol.2018180901.
14. LPIXEL Inc. [Internet]. Tokyo; LPIXEL Inc.. https://lpixel.net/en/. Accessed 8 Mar 2020.
15. Cover T, Hart P. Nearest neighbor pattern classification. IEEE Trans Inf Theory. 1967. https://doi.org/10.1109/tit.1967.1053964.
16. Cortes C, Vapnik V. Support-vector networks. Mach Learn. 1995. https://doi.org/10.1007/bf00994018.
17. Quinlan JR. Induction of decision trees. Mach Learn. 1986. https://doi.org/10.1007/bf00116251.
18. Ben-Bassat M, Klove KL, Weil MH. Sensitivity analysis in bayesian classification models: multiplicative deviations. IEEE Trans Pattern Anal Mach Intell. 1980. https://doi.org/10.1109/tpami.1980.4767015.
19. Bengio Y, Courville A, Vincent P. Representation learning: a review and new perspectives. IEEE Trans Pattern Anal Mach Intell. 2013;35(8):1798–828.
20. Hornik K, Stinchcombe M, White H. Multilayer feedforward networks are universal approximators. Neural Netw. 1989. https://doi.org/10.1016/0893-6080(89)90020-8.
21. Lecun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proc IEEE. 1998. https://doi.org/10.1109/5.726791.
22. Murakami K, Kido S, Hashimoto N, Hirano Y, Wakasugi K, Inai K. Computer-aided classification of diffuse lung disease patterns using convolutional neural network. Int J Comput Assist Radiol Surg. 2017;12(1):138–9.
23. Noguchi T, Higa D, Asada T, Kawata Y, Machitori A, Shida Y, et al. Artificial intelligence using neural network architecture for radiology (AINNAR): classification of MR imaging sequences. Jpn J Radiol. 2018;36(12):691–7.
24. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. In: NIPS'12: Proceedings of the 25th international conference on neural information processing systems, vol. 1. 2012. pp. 1097–105.

25. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:14091556. 2014.
26. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016:770–8.
27. Lee JH, Kim KG. Applying deep learning in medical images: the case of bone age estimation. Healthc Inform Res. 2018. https://doi.org/10.4258/hir.2018.24.1.86.
28. Poplin R, Varadarajan AV, Blumer K, Liu Y, McConnell MV, Corrado GS, et al. Prediction of cardiovascular risk factors from retinal fundus photographs via deep learning. Nat Biomed Eng. 2018. https://doi.org/10.1038/s41551-018-0195-0.
29. Girshick R, Donahue J, Darrell T, Malik J. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2014:580–7.
30. Girshick R. Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision. 2015:1440–8.
31. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu C-Y, et al. SSD: Single shot multibox detector. In: Leibe B, Matas J, Sebe N, Welling M, editors. Computer vision – ECCV 2016. ECCV 2016. Lecture notes in computer science, vol. 9905. Cham: Springer; 2016. pp. 21–37.
32. Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016:779–88.
33. Teramoto A. Dei-pu Ra-ningu [Deep Learning]. In: Fujita H (2019). Iryou AI to Dei-pu Ra-ningu Shiri-zu: Iyou Gazou Dei-pu Ra-ning Nyuumon [Medical AI and Deep Learning Series: Introduction to Medical Image Deep Learning]. 1st ed. Tokyo: Ohmsha, pp. 26–40. Japanese.
34. Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2015:3431–40.
35. Badrinarayanan V, Kendall A, Cipolla R. Segnet: a deep convolutional encoder-decoder architecture for image segmentation. IEEE Trans Pattern Anal Mach Intell. 2017;39(12):2481–95.
36. Ronneberger O, Fischer P, Brox T. U-net: Convolutional networks for biomedical image segmentation. In: International conference on medical image computing and computer-assisted intervention. 2015:234–41.
37. Higaki T, Nakamura Y, Tatsugami F, Nakaura T, Awai K. Improvement of image quality at CT and MRI using deep learning. Jpn J Radiol. 2019;37(1):73–80.
38. Dong C, Loy CC, He K, Tang X. Image super-resolution using deep convolutional networks. IEEE Trans Pattern Anal Mach Intell. 2015;38(2):295–307.
39. Umehara K, Ota J, Ishida T. Application of super-resolution convolutional neural network for enhancing image resolution in chest CT. J Digit Imaging. 2018;31(4):441–50.
40. Umehara K, Ota J, Ishida T. Super-resolution imaging of mammograms based on the super-resolution convolutional neural network. Open J Med Imaging. 2017;7(4):180–95.
41. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, et al. Generative adversarial nets. In: NIPS'14: Proceedings of the 27th international conference on neural information processing systems, vol. 2. 2014. pp. 2672–80.
42. Nakata N. Recent technical development of artificial intelligence for diagnostic medical imaging. Jpn J Radiol. 2019;37(2):103–8.
43. Mirza M, Osindero S. Conditional generative adversarial nets. arXiv preprint arXiv:14111784. 2014.
44. Zhu J-Y, Park T, Isola P, Efros AA. Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE international conference on computer vision. 2017:2223–32.
45. Hinton GE, Salakhutdinov RR. Reducing the dimensionality of data with neural networks. Science. 2006;313(5786):504–7.
46. LeCun Y, Cortes C, Burges CJ. The MNIST database of handwritten digits [Internet]. New York; [publisher unknown]. http://yann.lecun.com/exdb/mnist. Accessed 8 Mar 2020.
47. Glorot X, Bordes A, Bengio Y. Deep sparse rectifier neural networks. In: Proceedings of the fourteenth international conference on artificial intelligence and statistics. 2011:315–23.
48. Yasaka K, Akai H, Kunimatsu A, Kiryu S, Abe O. Deep learning with convolutional neural network in radiology. Jpn J Radiol. 2018;36(4):257–72.
49. Hashimoto M. Seeing is Believing, Considering is Understanding, Predicting is Discovering. Lecture presented at: The 78th Annual Meeting of Japan Radiological Society. 2019 Apr 11–14. Lecture material [xlsx file on the Internet]. http://rad.med.keio.ac.jp/wp/wp-content/uploads/2019/04/DeepLearning_Excel_ver1.0.xlsx.Japanese. Accessed 8 Mar 2020.
50. Kobayashi Y. Deep Learning nyuumon: nyu-raru nettowaku sekkei no kiso [Introduction to deep learning: basics for designing neural networks] [video on the Internet]. Tokyo; Sony Network Communications Inc.; 2019 Feb 26 [reviewed 2020 Mar 8]; [18 min., 37 sec]. https://www.youtube.com/watch?v=O3qm6qZooP0,Japanese. Accessed 8 Mar 2020
51. Saito K. Gakusyuu ni Kansuru Tekunikku [Techniques for Learning]. In: Saito K (2016). Zero kara tsukuru Deep Learning: Python de manabu dei-pura-ningu no riron to zissou [Building Deep Learning from Scratch: The Theory and Implementation of Deep Learning in Python]. 1st ed. Tokyo: O'Reilly Japan, pp. 165–203. Japanese.
52. Tieleman T, Hinton G. Lecture 6.5-rmsprop: divide the gradient by a running average of its recent magnitude. COURSERA Neural Netw Mach Learn. 2012;4:26–31.
53. Duchi J, Hazan E, Singer Y. Adaptive subgradient methods for online learning and stochastic optimization. J Mach Learn Res. 2011;12:2121–59.
54. Kingma DP, Ba J. Adam: a method for stochastic optimization. arXiv preprint arXiv:14126980. 2014.
55. Ioffe S, Szegedy C. Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:150203167. 2015.
56. Keras: The Python Deep Learning library [Internet]. [place unknown: publisher unknown]. https://keras.io/. Accessed 8 Mar 2020.
57. Multimedia [Internet]. Santa Clara: NVIDIA Corporation; c 2020. https://nvidianews.nvidia.com/multimedia. Accessed 8 Mar 2020.
58. Google Colaboratory toha [About Google Colaboratory] [Internet]. [place unknown: publisher unknown]. https://colab.research.google.com/notebooks/intro.ipynb.Japanese. Accessed 8 Mar 2020.
59. Microsoft Azure. Invent with purpose [Internet]. Seattle: Microsoft; c 2020. https://azure.microsoft.com/en-us/. Accessed 8 Mar 2020.
60. Neural Network Console [Internet]. Tokyo; Sony Network Communications Inc. https://dl.sony.com/. Accessed 8 Mar 2020.
61. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:160304467. 2016.
62. CUDA Toolkit 10.2 Download [Internet]. Santa Clara: NVIDIA Corporation; c 2020. https://developer.nvidia.com/cuda-downloads. Accessed 8 Mar 2020.
63. NVIDIA cuDNN [Internet]. Santa Clara: NVIDIA Corporation; c 2020. https://developer.nvidia.com/cudnn. Accessed 8 Mar 2020.

64. Visual Studio Best-in-class tools for any developer [Internet]. Seattle: Microsoft; c 2020. https://visualstudio.microsoft.com/. Accessed 8 Mar 2020.
65. Anaconda | The World's Most Popular Data Science Platform [Internet]. Austin: Anaconda, Inc.; c 2020. https://www.anaconda.com/. Accessed 8 Mar 2020.
66. Project Jupyter | Home [Internet]. [place unknown]: Project Jupyter; c 2020 [updated 2020 Jan 28; cited 2020 Mar 8]. https://jupyter.org/. Accessed 8 Mar 2020.
67. Collette A and contributors. HDF5 for Python [Internet]. [place unknown: publisher unknown]; c 2014. https://docs.h5py.org/en/stable/. Accessed 8 Mar 2020.
68. Matplotlib: Visualization with Python [Internet]. [placce unknown]: The Matplotlib development team; c 2002–2018 [updated 2020 Mar 4; cited 2020 Mar 8]. https://matplotlib.org/. Accessed 8 Mar 2020.
69. Clark A and contributors . Pillow [Internet]. [place unknown: publisher unknown]; c 1995–2020. https://pillow.readthedocs.io/en/stable/. Accessed 8 Mar 2020.
70. pandas [Internet]. [place unknown: publisher unknown]. https://pandas.pydata.org/. Accessed 8 Mar 2020.
71. SciPy.org [Internet]. [placce unknown]: TSciPy developers; c 2020. https://www.scipy.org/. Accessed 8 Mar 2020.
72. SPYDER [Internet]. [placce unknown]: The Spyder Website Contributors; c 2018. https://www.spyder-ide.org/. Accessed 8 Mar 2020.
73. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: machine learning in Python. J Mach Learn Res. 2011;12:2825–30.
74. Cython: C-Extensions for Python [Internet]. [place unknown: publisher unknown]. https://cython.org/. Accessed 8 Mar 2020.
75. OpenCV [Internet]. [placce unknown]: OpenCV team; c 2020. https://opencv.org/. Accessed 8 Mar 2020.
76. Pydicom website. https://pydicom.github.io/. Accessed Feb 2020.
77. Hara T. Kankyou Kouchiku [Environment Construction]. In: Fujita H, Hara T (2019). Iryou AI to Dei-pu Ra-ningu Shiri-zu: Hyoujun Iyou Gazou no tame no Dei-pu Ra-ning Jissen Hen [Medical AI and Deep Learning Series: Standard Deep Learning for Medical Images–Practice ver.-]. 1st ed. Tokyo: Ohmsha, pp. 1–26. Japanese.
78. NVIDIA DIGITS [Internet]. Santa Clara: NVIDIA Corporation; c 2020. https://developer.nvidia.com/digits. Accessed 8 Mar 2020.
79. Roth HR, Lu L, Liu J, Yao J, Seff A, Cherry K, et al. Improving computer-aided detection using convolutional neural networks and random view aggregation. IEEE Trans Med Imaging. 2016. https://doi.org/10.1109/tmi.2015.2482920.
80. Shin HC, Roth HR, Gao M, Lu L, Xu Z, Nogues I, et al. Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. IEEE Trans Med Imaging. 2016. https://doi.org/10.1109/tmi.2016.2528162.
81. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res. 2014;15(1):1929–58.
82. Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE international conference on computer vision. 2017:618–26.
83. Binder A, Montavon G, Lapuschkin S, Müller K-R, Samek W. Layer-wise relevance propagation for neural networks with local renormalization layers. In: International Conference on Artificial Neural Networks. 2016:63–71.