The Role of Deep Learning in Improving Healthcare

Stefan Thaler and Vlado Menkovski

1 Introduction

Many industries including healthcare benefit significantly from advances in information technologies particularly with the growing trend of digitalization. Significant improvements in efficiency and reduction of costs are achievable through automation of processes, cost-effective storage, and efficient retrieval of information and data [122]. This cost reduction is particularly important in industries such as healthcare where rising costs and aging populations impose continuous pressure on existing practices. The need for improving medical procedures, improving prevention, early detection, and more efficient and more accurate diagnosis is evident. The role of artificial intelligence (AI) technologies in this context is particularly important. The promise of these technologies is to deliver an unprecedented opportunity for automation [82].

One AI technology that already creates a significant impact on healthcare is machine learning (ML). ML allows for the development of data-driven solutions to complex problems. These approaches deliver more scalable and robust solutions to existing expert designs, i.e., solutions that have been manually crafted by human domain experts. They can be developed and adapted much more efficiently than manually devised solution having a lot of data already collected. The predictive analysis enabled by ML can deliver solutions in clinical diagnosis, prevention, and healthy living.

However, in the realm of high-dimensional data (high-resolution imaging, high-precision patient monitoring, molecular testing), traditional ML algorithms face the curse of dimensionality [59] and have to rely on costly feature engineering by

S. Thaler (✉) · V. Menkovski
Eindhoven University of Technology, Eindhoven, Netherlands
e-mail: s.m.thaler@tue.nl; v.menkovski@tue.nl

S. Consoli et al. (eds.), *Data Science for Healthcare*,
https://doi.org/10.1007/978-3-030-05249-2_3

experts to deliver solutions. Recent developments in ML focus on approaches that use multiple layers of representations. These layered architectures allow learning representations that are useful for solving a task purely from data without the need for features that are crafted by a domain expert. These technologies that use multiple layers of representation can loosely be grouped as Deep Learning technologies, and form a subfield of ML. In this chapter, we will focus on Deep Learning (DL) technologies and their impact on healthcare.

We will present an overview of the underlying concepts and algorithms that drive the success of DL. We will motivate the use of DL on different types of data and discuss applications of DL to various fields in healthcare. DL is a data-driven technology. The way DL is applied mainly depends on the structure of the data. Even though in different applications the data indeed comes in a different context with specific semantics, we present a data-centric view of the structure and organize the applications of DL based on the type of data. We start by discussing spatially correlated data, which includes various imaging modalities ranging from radiology to digital pathology. Furthermore, we relate this to other spatially correlated data such as patient monitoring systems such as electrocardiography (ECG) or electroencephalography (EEG). Although they are time series, these data also carry spatial correlations; hence we treat it as 1D spatially correlated data. Then we present an overview of sequential data (temporal and other sequences) and natural language text.

This survey aims to consolidate a wide range of work in DL applications to healthcare with a data-centric perspective that brings insights into the maturity of the technology and its drawbacks and invites directions for future applications.

The remainder of this chapter is organized as follows. In Sect. 2, we motivate on a general level the use of ML for healthcare problems. In the same section, we proceed to motivate DL by pointing out a few shortcomings of ML. In Sect. 3, we briefly describe major building blocks for DL methods, and in Sect. 4, we outline general strategies on how to use such building blocks to solve problems. Sections 3 and 4 describe the background knowledge that is necessary to understand the main contents of this book chapter, but a proficient reader may safely skip them. Section 5 describes application of DL technologies on healthcare problems. Section 5 is structured into three subsections, each of which presents an overview of state-of-the-art approaches for a data modality, i.e., Sect. 5.1 describes approaches on sequential data, Sect. 5.2 describes approaches on spatial data, and Sect. 5.3 describes approaches on text data. Each of these three subsections follows the same structure: first, the data modality and the sources of this data modality are introduced; then, we present an overview of approaches grouped by the problem that they are solving. We link the approaches to advancements in other fields such as recognition and outline characteristics of the architectures that were used. We conclude this chapter with a summary and pointers for future research in Sect. 6.

2 Learning from Data

In data analysis, there are some tasks which are difficult to solve with computer algorithms. For example, it is very challenging to create an algorithm for detecting or segmenting organs in a CT scan. Such tasks are challenging because their creation requires a deep understanding of the domain, and often complex relationships in the data are not fully understood.

Expert systems such as CADUCEUS [7] are one way to address such tasks. Expert systems are rule-based systems that emulate human experts and attempt to solve a task by evaluating a set of rules about the data. Expert systems have their important role in many applications particularly when it is critical to have graceful degradation of performance. Furthermore, expert systems clearly explain the decision, which is vital for domains where accountability is important, e.g., healthcare, information security, or law enforcement. Expert systems are useful. However, they have a few caveats. First, designing rules is difficult and time-consuming. To devise good rules, domain experts need to understand the domain and the data very well. They need to adapt to evolving contexts, and often domain knowledge requires to produce a large number of exceptions for each rule. Moreover, handcrafted rules are often brittle, and their maintenance is costly, mainly when the expert system contains many, potentially conflicting rules.

Another possibility of addressing such difficult data analysis tasks are data-driven approaches, which offer the potential to build models purely from observation. Here machine learning algorithms allow for developing such models by processing available observations or data. There is a vital role for such algorithms in healthcare since in many domains the underlying processes are not fully understood particularly in medicine. Another aspect is noisy measurements that may require observing data to extract the useful information using machine learnings.

Machine learning can broadly be categorized into three categories: supervised learning, unsupervised learning, and reinforcement learning. In a supervised learning setting, data are associated with one or more targets, and a model is learned to predict such associations. For example, to categorize nuclei images, the pixels of the image are the data, and the different nuclei are the targets. In healthcare, many problems such as clinical decision support, image segmentation, or image registration have been addressed in a supervised way.

Unsupervised learning attempts to discover patterns in the unlabeled data. Such patterns can be used, for example, to learn more suitable representations, to compress the data, or to find cohorts in data. In healthcare, unsupervised learning is used primarily for learning useful features, e.g., if the dimensions of the input space are too large. In the context of DL, the algorithms themselves for unsupervised and supervised learning cannot clearly be distinguished, i.e., an unsupervised learning algorithm will also learn model parameters by optimizing for specific targets. The critical difference is that these targets have not been labeled by some external agent.

In reinforcement learning, an agent interacts with an environment in a feedback loop and attempts to learn to complete a task. Reinforcement learning is also applied in healthcare, e.g., [72]. However, in this chapter, we will omit reinforcement learning since the developments are very recent and in a relatively small number. In the future, reinforcement learning may play a greater role in healthcare, e.g., for drug design or autonomous health support agents.

Early machine learning methods devised handcrafted features from the input data and learned predictive, so-called "shallow" models for these input features. Shallow machine learning methods have been hugely successful in many application domains, but they have a few shortcomings. First, they require domain experts to devise sensible features for the task at hand. These handcrafted features suffer from the same limitations as rule-based systems: they are brittle, domain-specific, and labor expensive and challenging to create.

Another major limitation of shallow algorithms is working with high-dimensional data. When the dimensionality grows, particularly when the ratio of features to data points becomes low, many shallow machine learning algorithms perform poorly. This problem is known as the curse of dimensionality and results in the machine learning model to overfit, i.e., the inability to generalize well.

DL enables a high level of generalization when working with high-dimensional data. It relies on models built with artificial neural networks, which process the data in a sequential fashion and allow for creating composite features, starting from low-level to high-level features in a hierarchical manner. This process is referred to as representation learning and enables this model to be successful in this type of data.

Furthermore, DL methods deal well with noisy data. In fact, noisy data enables DL models to learn better generalizations, because it is assumed that data lies on a lower-dimensional manifold. The noise in the data helps discover this manifold.

Another significant advantage of deep neural networks is computational efficiency. Composed hidden layers result in exponentially less required training steps to achieve good generalizations [111]. Furthermore, stochastic gradient descent enables to train on very large datasets effectively.

Finally, DL methods learn features that are useful for the task at hand from data, which reduces the need for domain-specific feature engineering. Input features to DL algorithms are generally domain independent and impose very few assumptions on the input data. Consequently, architectures that work for data types in one domain can readily apply in other areas which face different problems, but a similar data type.

3 Deep Learning Methods

Similar to many machine learning algorithms, DL algorithms consist of four components: data, a learning objective, a model, and a training procedure. The right combination of these four components is essential for solving a task using DL techniques.

In this section, we will only briefly outline building blocks of core DL models. We start by introducing a single artificial neuron. We'll elaborate the interaction between the model, the data, the objective, and the training procedure on such an artificial neuron. Artificial neurons are not deep models, but they are a fundamental building block of deep neural networks. We then use multiple artificial neurons to create a layer, and we compose feed-forward neural networks out of multiple of such layers. Using a combination of such layers and parameter sharing, we then introduce recurrent neural networks and convolutional neural networks, which are well suited for data with sequential and spatial correlation. We conclude this chapter with generative models, which use layers of neural networks to learn the data-generating distribution.

3.1 Artificial Neuron Model

An artificial neuron is a parametric function that is very loosely inspired by the way neurons in human beings work. Artificial neurons are a fundamental building block of modern deep neural networks. A human neuron receives inputs from multiple other neurons. If the stimuli of the neural cell surpass a certain threshold, the neuron will pass a signal to the other cells that it is connected with via its Axon.

Similarly, artificial neurons receive "stimuli" from the input data. The artificial neuron processes these stimuli, and if they surpass a certain threshold defined by the activation function, the neuron "fires." Figure 1 depicts a schematic overview of such a neuron. The pixels of the image are treated as input stimuli to the neuron; the neuron processes the input stimuli by multiplying them with a weight for each input and summing up the result. It decides to fire if the activation function returns a value larger than zero, and it does not fire otherwise.

$$y = f(x; W) \tag{1}$$

A single neuron with the logistic function as activation is similar to logistic regression [95]. For DL methods, artificial neurons only play an important role as building block for more sophisticated architectures, such as fully connected layer.

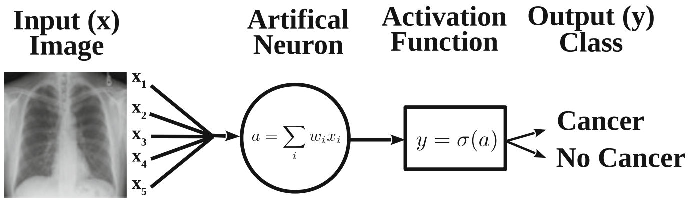

Fig. 1 The schematic architecture of an artificial neuron. The input values are multiplied by weights and summed up to a. The output of the neuron is the result of the nonlinear activation function σ of a. The X-ray image is a courtesy of Wikimedia Commons

Any smoothly differentiable function can be an activation function. Smooth means with continuous derivatives within a certain domain to a desired order. In some cases activation functions may also be non-smooth, e.g., in case of the ReLU, which use sub-gradients for solving gradient-based optimization problems. However, certain properties are desirable. The activation function should be monotonic so that the error surface remains convex. It should be nonlinear so that more complicated functions can be approximated. Popular activation functions are ReLU [84] (ConvNet, RBM), TanH (LSTM), Softmax (MultiClass, Single Label), and Sigmoid (Multi-Class, Multi-Label).

3.1.1 Objective Function

In DL, the goal is to complete a task using a model. The available data "teaches" the model the parameters. These parameters should be chosen in such a way that they enable the model to complete the task in the best possible way. However, one needs to define what the best possible way is. In DL, this is achieved by the objective function or cost or goal function. The objective is a function defined over the output of the model and tells it how wrong its prediction was based on the ground truth information $\hat{y}$ for that example. This information can then be used to minimize such wrong predictions; hence find a model that approximates the desired target function.

$$c = J(f(W; x), \hat{y})$$

Objective functions have different properties, which significantly impact the outcome of the DL procedure. If the objective function is convex such as the mean squared error (MSE), a global minimum can be found. Otherwise, it is possible only to find a locally optimal solution. Popular objectives are the MSE for regression tasks, binary cross-entropy for binary classification problems, and categorical cross-entropy for multiclass, single-label classification tasks.

3.1.2 Training Artificial Neurons

Our goal is to find an approximate model that allows us to solve a task at hand. To do so, we need to find the parameters that approximate the model in the best possible way defined by the data. Within DL, parameters are almost exclusively learned by back propagation [107] and variants of mini-batch stochastic gradient descent [107].

The back propagation algorithm consists of three main steps: a forward pass, a backward pass, and a parameter update. The forward pass calculates the cost of given input examples concerning the objective using a model with current parameters. In other words, the forward pass calculates how wrong the model with current parameters is. In the backward pass, the partial derivatives of the model's parameters are computed with respect to the objective function. This backward pass tells how much a parameter influences the cost of the result. In the final step, the parameters get updated using the partial derivatives from the parameter update multiplied by a learning rate. The learning rate ensures that we progress along to a minimum of the objective along the error surface.

Ideally, the parameters are calculated for all available training data. However, many datasets these days are large, which renders this procedure computationally impractical. Instead, parameter updates are computed for a small subset of the data. Such a subset is often called a batch or a mini-batch. If the learning rate is well-chosen, mini-batch stochastic gradient descent will eventually find a minimum of the cost function—either a local minimum or in case of a convex cost function a global minimum.

To speed up the parameter learning process, multiple momentum-based variants of mini-batch stochastic gradient descent have been proposed. Such variants will increase or decrease the learning rate depending on the stage of the learning process or some properties of the data. Popular variants of stochastic gradient descent are Adagrad [29], RMSProp [123], and Adam and Adamax [61].

3.2 Deep Feed-Forward Neural Network

Feed-forward neural networks are neural networks that are composed of layers of artificial neurons. This composition allows each layer to use the features of the previous layer to create more abstract features. Such a network learns to produce features that are helping to solve the task at hand.

More formally, a feed-forward neural network is a nonlinear, parameterized function that is composed of multiple, nonlinear parameterized functions. These various functions are commonly referred to as layers. This function maps input data x to output data y in such a way that it approximates the desired function for the task at hand in the best possible way.

For example, if the task is cancer detection in X-ray images, then the feed-forward neural network is a function that maps the input pixels of the X-ray images to two classes, cancer or no cancer.

Each layer of a feed-forward neural network is a function that maps input x to some output h in a nonlinear way. Commonly, layers of feed-forward neural networks have three components: parameters W, biases b, and a nonlinear function σ that is referred to as activation function.

The number of neurons (often called neural units or simply units) per layer determines the output dimension of representation that is learned by this layer. For example, if a layer has 20 units (or neurons), the representation that is learned by this layer is 20-dimensional. The more neurons a layer has, the more capacity it has to describe the input data. However, if a layer has too many units, it will start overfitting the data, i.e., learn to memorize the training data. If a layer has too few neurons, it will begin underfitting, i.e., generalizing too much. This problem is known as bias-variance problem [34] in the machine learning community and is also applicable to DL.

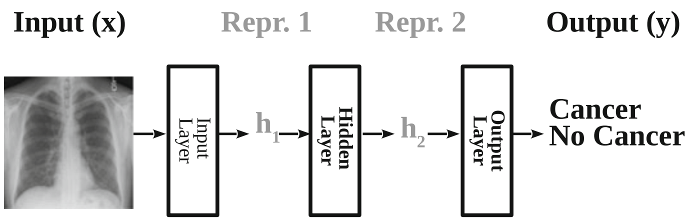

Fig. 2 Schematic of a feed-forward neural network with one hidden layer for X-ray image classification. The pixels of the X-ray image get mapped to a hidden representation, which in turn gets assigned to another hidden representation, which then gets mapped to the classes. The X-ray image is a courtesy of Wikimedia Commons

Deep neural networks learn distributed representations [41]. Distributed representations are compelling because they potentially can express an exponential amount of data. For example, a binary, k-dimensional representation can represent up to 2^k data samples, as each dimension of the representation can store associations of the data independently. An example of non-distributed representations is one-hot vectors. A k-dimensional one-hot vector can only represent k examples (Fig. 2).

A feed-forward network is defined by a model that maps an input x to an output y. They are composed of multiple layers: an input layer, one or many hidden layers, and an output layer. The input layer represents the function from the input data x to the first intermediate output h_1, the output layer a mapping from the last intermediate result h_n to the output data y, and the hidden layers map one intermediate result h_{n-1} to another intermediate result h_n. The output of the intermediate layers is unknown a priori; therefore they are commonly referred to as hidden layers. The neural network chooses the output of the hidden layer in such a way that it approximates the desired function defined by the learning objective well.

Feed-forward neural networks can be used on any data and, given sufficient capacity, can learn arbitrary functions [46]. Intuitively, this makes sense because a single-layer model with sufficient capacity will learn to map one input to one output. Neural networks that are composed of multiple layers learn such a mapping more efficiently by generalizing.

However, feed-forward neural networks come with several limitations. Firstly, the number of parameters to train a deep feed-forward neural networks is potentially very high since the input of each layer is connected to all outputs of the layer. Another limitation is that the input dimension is fixed. For example, if a feed-forward neural network is used to classify images, the input images all need to have the same dimension. Thirdly, feed-forward neural networks tend to overfit the data [34], and finally, it may take a long time for the model's parameters to converge.

Purely deep feed-forward neural networks are rarely used to solve healthcare challenges. Instead, they are often part of a more sophisticated architecture, such as convolutional neural networks (see Sect. 3.3), recurrent neural networks (see Sect. 3.4), or autoencoders (see Sect. 4.1).

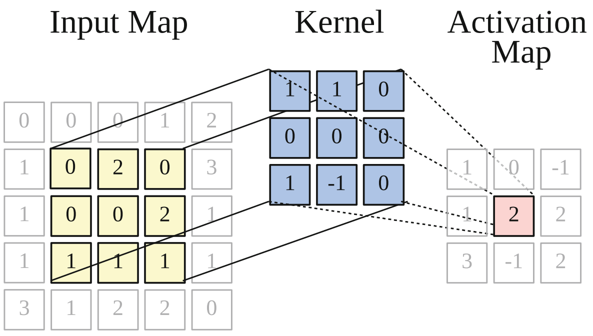

Fig. 3 Schematic overview of the convolution layer. A local receptive field "slides" over the input to create an activation map. Each convolution operation shares the parameters of that particular receptive field

3.3 Convolutional Neural Networks

Feed-forward neural networks make minimal assumptions about the data that they are processing. However, often we have general information about the data that we are processing. One example of such data is images. Pixels in images usually have a loose spatial correlation, e.g., consider an image of a tree. If a pixel of the image represents the tree's bark and has a brown color, the pixels close to that pixel are also a bit more likely to be brown.

Convolutional neural networks (CNNs) are a particular type of feed-forward neural networks that use this spatial information correlation to design neural networks that perform better at processing such data. CNNs combine three key ideas: local receptive fields, parameter sharing, and local subsampling.

Local receptive fields, also called kernels, connect small patches of the input data with one point of the output data. Local receptive fields assume that the input data are spatially correlated, i.e., that the neighborhood of a data point influences this data point and vice versa. Fully connecting all small patches with the outputs is computationally impractical as it drastically increases the number of parameters. Instead, parameters for such local receptive fields are "slid" over the input, and an output is calculated for each different position. The parameters are shared for each position. When training CNNs, multiple kernels per convolutional layer will be trained and slid over the input, thereby producing multiple activation maps. This approach drastically reduces the number of parameters needed. Figure 3 depicts schematically how local receptive fields and shared parameters are used to create an activation map.

Networks that are composed of multiple convolutional layers also require a large number of parameters to learn. To reduce the computational strain, subsampling layers, also called pooling layers, can be inserted. Such pooling layers reduce the spatial size of the activation map by pooling multiple locally connected values to a single value. Various such pooling strategies have been proposed. Two popular approaches are max pooling and average pooling. In max pooling, the maximum of a local receptive field is passed on, and in average pooling, the average of the values of the receptive field is passed on. Alternatively, one can subsample the network using convolutions with local receptive fields of width and height of one [120]. These 1×1 convolutions will have a subsampling effect, but instead of choosing the best value of a local receptive field, they will select the best available one.

A CNN generally consists of multiple convolutional and pooling layers. This chaining of layers results in a hierarchical structure of the locally receptive fields. Consequently, the more layers such a network has, the larger the total receptive field of the network is. The network will use this hierarchy of features to represent more high-level concepts.

Since CNNs are a special case of feed-forward neural networks, they are commonly trained similarly by backpropagation and a variant of mini-batch stochastic gradient descent. Also, often last layers of CNNs are fully connected layers.

CNNs work well on data that is locally spatially correlated. They do not necessarily require two-dimensional inputs but also work on one or more dimensional input, as long as there is a local, spatial correlation. An example for 1D locally correlated data is ECG signals, and an example for 2D locally correlated data is X-ray images.

CNNs bear advantages over plain feed-forward neural networks when applied to spatially, locally correlated data. Firstly, they require far fewer parameters to be trained and are therefore much more computationally efficient than feed-forward neural networks. Secondly, the subsampling operations lead to a particular shift, scale, and distortion invariance of the learned model. Another advantage is that they work on input data of arbitrary size. Sliding kernels on differently sized input result merely in differently sized activation maps. Finally, many applications demonstrated that CNNs are well suited for transfer learning [129], i.e., they are trained on a large, generic image dataset of one domain and fine-tuned on a small dataset in another area.

Modern CNNs may consist of more than 100 layers (e.g., [40]) and have many hundreds of millions of parameters, in extreme cases even billions of parameters [108]. Such large models are costly to train and need lots of training data to converge. Finally, they do not perform well on data which is not locally correlated.

An example application of CNNs in healthcare is presented in the work of Shen et al., who use CNNs to predict whether lung nodules are malicious or not [110].

3.4 Recurrent Neural Networks

Another generic assumption one can make about the data is temporal (or sequential) interdependence of data, as time series, natural language, or sound. If the data is locally sequentially correlated, 1D CNNs can be used to learn representations from such patterns. For more complex patterns or patterns that occur over time, recurrent neural networks (RNNs) have been developed.

RNNs are neural networks that are designed in a way to reuse the outputs of the network in later calculations, i.e., in a recursive way. Similar to CNNs, RNNs rely on parameter sharing, but in a different fashion. In addition to parameter sharing,

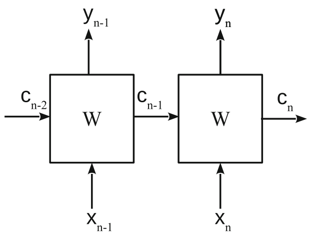

Fig. 4 Schematic overview over an RNN. It processes a sequence of inputs and consists of two functions: one that yields the next state and one that generates the output of the current network. The parameters are shared over the whole sequence of inputs

RNNs remain a state (or context) of the network, which they pass on for further processing.

RNNs require the input sequence to be discretized into a series of time steps. For each of the time steps, the current input and the context of the previous time step are used to calculate the output of the network and the next state. Two functions are learned: one that yields the output of the current time step and one that produces the context for the next time step. The parameters of the network are shared over the whole sequence. The state maintains a "memory" of which inputs have been processed so far. Figure 4 schematically depicts the processing steps of an RNN.

RNNs can be used to learn functions in flexible ways. They can be used to learn functions to map sequences to a single output (many-to-one), for example, to classify sequences. They can be used to learn functions that map sequences to other sequences (many-to-many), for example, to tag sequences with specific labels or for natural language translation [16]. And they can be used to map a single value to multiple outputs (one-to-many), for example, to generate descriptive text from an input image [128].

Due to the feedback connections, RNNs cannot be trained using the standard backpropagation algorithm. Instead, an extended algorithm is used—backpropagation through time (BPTT) [130]. BPTT follows the three basic steps of back propagation: forward pass, backward pass, and parameter update. To do this, the neural network is unrolled for n time steps. That is, the parameters are replicated n times, which allows the outputs and contexts for the forward pass to be calculated. Then, the gradients are calculated for each time step individually in the backward pass. The gradients are averaged by the n. Finally, the parameters are updated with these averaged gradients.

Theoretically, RNNs can be used to learn functions that deal with sequences of an arbitrary length. In practices, however, two problems occur when the processed sequences are too long. First, the gradient that flows back throughout the time gets very small, so that the network stops learning, which is also referred to as vanishing gradient problem. To overcome these problems, long short-term memory (LSTM) networks have been proposed [43]. They introduce trainable gates that learn when to forget and when to pass on gradients. The other problem that commonly occurs when training RNNs for longer networks is exploding gradients. That is, the network stops learning because some of the gradients get excessively large over time. To counter exploding gradients, gradient clipping has been proposed [93], which truncates gradients if they surpass a certain threshold.

RNNs are versatile DL models. They can be used on sequential data of arbitrary length, and they are capable of capturing complex, time-dependent relationships within the sequential data.

However, training them may be difficult, and it may require large amounts of data to converge. Also, unrolling them for many time steps requires the parameters to be replicated many times, which is computationally very expensive.

In healthcare, RNNs are commonly applied to solve problems on sequential data. An example for such an application is to predict seizures from raw EEG data [83].

3.5 Autoencoders

Autoencoders are composite models that consist of two components: an encoder model and a decoder model. The task of an autoencoder is to output a reconstruction of the input under certain constraints. Figure 5 depicts this general architecture schematically. The encoder and the decoder model can be any neural network, preferably one that works well with the data type. So one can imagine an autoencoder for images where the component models are CNNs or an autoencoder for text where the component models are RNNs.

If autoencoders have sufficient capacity, they will learn two functions that will simply copy the input to the output. Such functions are generally not useful. Therefore the representations that the autoencoder has to learn are typically constrained in specific ways, for example, by sparsity or by a form of regularization. Such constraints force the encoding model to learn representations that contain potentially useful properties or regularities of the data.

Autoencoders can be used for many tasks. One such task is that they can be used to learn representations in an unsupervised way. To do so, an autoencoder is trained, and after that, the encoding model is used to derive the representations from the input data. Such a representation can be thought of as a nonlinear dimensionality reduction of the input. Another task is to denoise input data. To do so, the input

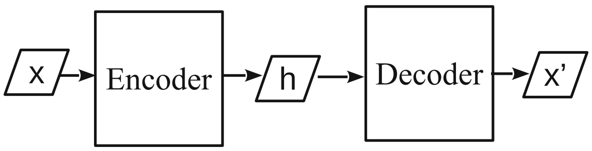

Fig. 5 Schematic of an autoencoder. An autoencoder consists of two models, an encoder and a decoder. The encoder maps an input x to a representation h, and the decoder reconstructs x given h. Encoder or decoder can be any neural network

of the encoder is distorted by some noise, e.g., Gaussian noise, and the target of the autoencoder is learning to reconstruct the original input and thereby learning to remove certain distortions from the data.

Autoencoders can be trained in an unsupervised fashion since both the input and the target output are both the data x. If an autoencoder is used to combine many sparse layers, then each of the layers is trained one after the other. For example, let x_n be the input of the n-th layer, $\hat{x_n}$ the output of the n-th layer, l_{en} the n-th encoding layer, and l_{dn} the n-th decoding layer of the autoencoder. Then the first training step for an autoencoder is to learn the functions l_{e1} and l_{d1} such that $\hat{x_1} = l_{d1}(l_{e1}(x_1))$ and the loss $l_1 = L(x_1, \hat{x_1})$ are minimal to a given loss function L. Then the parameters of the functions l_{e1} and l_{d1} will be fixed, and the next layer's function will be trained in a similar fashion as the first layer, but by using $\hat{x_1}$ instead of x as input and target output.

Autoencoders may be used for a variety of useful tasks such as enhancing input data quality or learning to compress input data in a meaningful way. Autoencoders can be trained in an unsupervised fashion, which may help to solve problems where there is a significant amount of unlabeled data and labeling data is scarce and typically costly to obtain. An example application for autoencoders for healthcare is to predict the future of patients by learning suitable representations from electronic healthcare records [81].

3.6 *Generative Models*

From a probabilistic perspective, the DL methods that we discussed in the previous sections learn the conditional distribution $P(y|x)$, i.e., the likelihood of seeing the output y given an input x. In contrast to that, generative models aim to learn the data-generating distribution $P(x, y)$ from the data, i.e., how likely it is to see both x and y at the same time. Knowing the joint distribution allows to predict specific outputs given an input and also to generate new data from the learned model.

There are three major types of DL methods that are used to train generative models: deep belief networks [9, 42], variational autoencoders [62], and generative adversarial networks [37]. An example for generative models in healthcare is image synthesis to increase the amount of available training data and thereby improve existing data analysis methods [89].

3.6.1 Deep Belief Networks

Deep belief networks (DBNs) [42] were among the first deep, generative models. They are directed, probabilistic graphical models. They are composed of multiple layers of restricted Boltzmann machines (RBMs) [114]. RBMs are energy-based models that learn a joint probability distribution of the input and output data. This joint probability distribution is defined by an energy function. Figure 6 depicts a

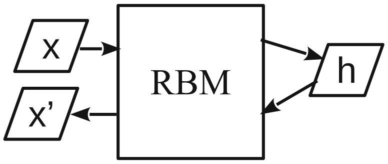

Fig. 6 Schematic of a RBM. The network learns $P(x, h)$ by learning $P(h|x)$ and then $P(x|h)$. DBNs are composed of multiple RBM layers

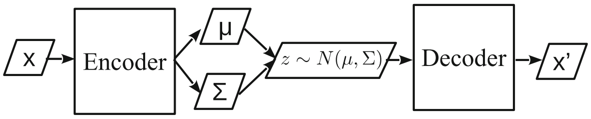

Fig. 7 Overview over a variational autoencoder (VAE). An encoder model learns μ and Σ of a multivariate Gaussian given x. A decoder model learns to predict x given a random value z that was sampled from this Gaussian. The decoder network is the generative model

schematic overview over a RBM. The restriction in the RBM is that there are no intra-layer connections in the hidden layer.

RBMs learn the joint probability distribution $P(x, y)$ of the input x and output y by first learning the conditional distribution $P(x|y)$ of output y given input x and then learning the conditional distribution $P(x|y)$ of input x given output y. DBNs are trained by using multiple layers of RBMs after each other in a similar way autoencoders are trained. DBNs and RBMs have an intractable partition function, which means that they need to learn an approximation.

Among others, deep belief models and RBMs can be used for dimensionality reduction or data sampling. For example, DBNs have been used to learn suitable representations from microarray data to predict breast cancer [60].

3.6.2 Variational Autoencoders

Similar to autoencoders, variational autoencoders (VAEs) consist of two models, an encoder model and a decoder model. The encoder model learns Σ and μ of a multivariate Gaussian distribution given a certain input x. This distribution is used to sample a random variable z. The decoder model learns to reconstruct x given z. The decoder model is the generative model. Figure 7 depicts the architecture of a VAE.

3.6.3 Generative Adversarial Networks

Generative adversarial networks (GANs) also consist of two models, a generator model and a discriminator model. The input to the generator model is a random

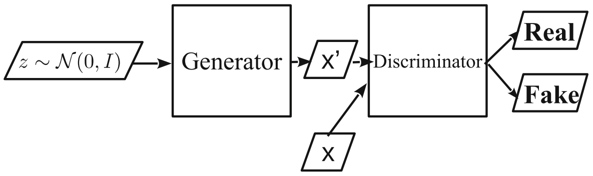

Fig. 8 Overview over a generative adversarial network (GAN). A generator model learns to generate inputs x' from z that was sampled from a multivariate uniform Gaussian. A discriminator model is either presented a real input x or a generated x' and has to decide which one is real and which one generated

variable z that is sampled from a multivariate Gaussian distribution. The task of the generator model is to generate an input x given z. The discriminator model is trained on a binary classification task, namely, to distinguish if an input x is real or fake input. Figure 8 depicts the architecture for a generative adversarial network.

3.6.4 Training of Generative Models

VAEs and DBNs can be trained in an unsupervised way and with standard mini-batch SGD. GANs are trained with a game-theoretical approach where the generator model tries to beat the discriminator model. In its simplest form, training GANs resembles a two-player minimax game, where each player attempts to maximize its own value function. Training GANs may be challenging in practice due to problems such non-convergence [36]; however many improvements have been suggested, e.g., [2, 105].

3.7 Other Methods

There are many more existing DL methods and many recent developments such as deep reinforcement learning, capsule networks [104], or neural Turing machines [38] or architectures for metric learning such as Siamese [21] or triplet networks [45]. Outlining them is out of the scope of this chapter, but [10] and [107] provide an overview of the field.

4 Data Analysis Strategies Based on Deep Learning

In this section, we describe abstract solutions for common DL problems. DL problems usually consist of a task to be solved, an objective function, a model architecture, and data which is used to learn the parameters of the model. The model architecture and the objective function commonly depend on the nature of the data that is analyzed. However, the strategy to address a problem can often be reused for other problems. For example, consider the task of organ segmentation. This task can be framed as a classification task where each pixel is categorized as belonging to the organ or not belonging. The model architecture and objective function depend on the data, e.g., CT scans. However, the strategy of classifying pixels can be used for other tasks as well.

In this section, we will use x as the vector of input features, y as ground truth target, and $\hat{y}$ as the predicted target that was derived by the DL solution. Furthermore, f will be the parameterized DL model that we are attempting to learn. f maps input x to the predicted target $\hat{y}$, i.e. $\hat{y} = f(x)$. We refer to $C(y, \hat{y})$ as our objective function that defines a measure of correctness of our prediction.

4.1 Representation Learning

DL architectures are usually composed of multiple layers of nonlinear functions. Layers closer to the input learn representations that are useful for next layers to minimize the objective function for that problem. As a consequence, multilayer architectures always learn representations to solve a particular task. For example, in a classification task, the representations that are learned by a multilayer network are used to perform that classification. However, sometimes it is desirable to learn such features (or representations) directly. For example, if the input data is high dimensional, it is desirable to learn a lower-dimensional representation for another model to be able to solve a task. There are two common strategies of representation learning using DL technologies—using unsupervised methods such as autoencoders and transfer learning approaches.

The general strategy on how to learn representations using autoencoders is depicted in Fig. 9. First, the autoencoder is trained in an unsupervised way. After training, only the encoding function is used to derive representations from data. We have described autoencoders in more detail in Sect. 3.5.

A variant of unsupervised representation learning is stacked autoencoders. Generally, deep autoencoders consist of two deep neural networks that are trained in an end-to-end fashion. Stacked autoencoders are also deep models, but their layers are trained one at a time. Figure 10 depicts this approach. Stacked autoencoders are useful for solving large problems where the whole model would go beyond the

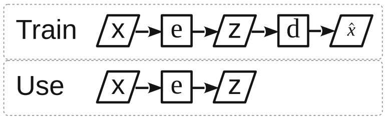

Fig. 9 Overview of how to learn representations using autoencoders. During training, an encoder e maps the input x to the representation z and a decoder d attempts to reconstruct the input x given z. After training, the decoder is discarded, and the encoder e is used to derive the learned input representations

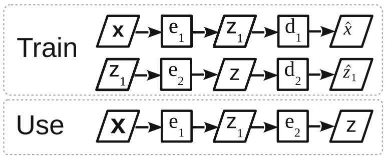

Fig. 10 Overview of how to learn representations using stacked autoencoders. Each layer is trained to reconstruct its input. After the training, the decoding layers are discarded and the encoding layers chained to derive the learned representation

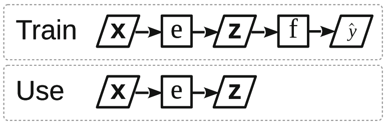

Fig. 11 Overview transfer learning for representation learning. A network consists of an encoding part e and a domain- and task-specific model f. After training, f is dropped and representations are derived using e. Any multilayer neural network can be considered a combination of a decoding part and a domain-specific model

scope of the computational resources available. Deep belief networks [42], which are composed of multiple layers of RBMs, were among the first architectures that were successfully trained for deep representation learning.

Representation learning can be used for many useful tasks, such as compressing or denoising the input. In healthcare, representation learning using autoencoders has been applied to a broad variety of tasks. A selection of these tasks is learning representations from EHR to predict diseases [81], reducing the input dimensionality for gene expression profiling [32], and learning features from breast images to predict cancer risk [54].

Another way to learn representations is in a supervised fashion using transfer learning. Transfer learning is based on the idea that any multilayer neural network can be represented as two parts: an encoding part and a task-specific model. Both parts can be multilayer and are not restricted to any particular architecture. If such a model is trained with a sufficiently large corpus of data, then the features learned in the encoding part should generally apply to other problems. Figure 11 depicts the general strategy for transfer schematically.

In healthcare, representations obtained by transfer learning have been mainly applied to medical image analysis. There, large models such as VGGNet [113] or ResNet [40] were trained on general image corpora such as image net on a classification task. In many cases, all layers except the last classification layer were considered to be the encoder model for deriving the image features. These generic image features were then used for domain-specific tasks, for example, Esteva et al. used a pre-trained Inception V3 architecture to detect skin cancers [31].

4.2 Classification

Classification is the task of assigning one or multiple discrete labels y to a given input x. The general strategy to solve such a problem is to learn a function that maps the input to the output $\hat{y} = f(x)$, where f can be any architecture that is suitable for the data at hand.

In healthcare, many challenges can be solved by treating them as classification problems. For example, lesion segmentation can be framed as binary classification problem by learning a function that for each pixel of the input image predicts whether it is a boundary pixel of a lesion or not, e.g., [103]. Another example of classification in the medical domain is nodule classification. There, the input is an image of an object, and the DL network needs to decide whether the shown object is a nodule or not, e.g., [109]. A third example of a classification task in healthcare is learning word vectors from electronic health records. There, the task is to predict a word given a context of other words, i.e., the words are considered the classes that can be chosen for a given input [18].

Another task that is related to classification and can be solved with similar strategies is regression. Regression differs from classification only by the output. That is, in classification commonly you have discrete outputs, whereas in regression the output can be real-valued. In healthcare, regression is often used for registration tasks such as [79], where a CNN is used to regress the spatial alignments between two X-ray images.

4.3 Anomaly Detection

Anomaly detection is the task of finding outliers in data. One strategy to detect anomalies using DL is to train a model to learn to predict "normal" values, where normal is defined by your training data. Anomalies are then detected when for a test case the predicted data deviates more than a specified threshold from the actual data. We depict the general strategy for anomaly detection using DL in Fig. 12. The model f is agnostic to the architecture and again should be chosen to fit the peculiarities of

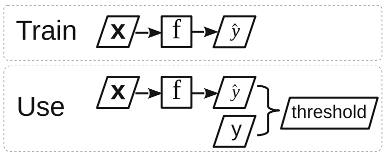

Fig. 12 Overview of the anomaly detection task. Anomalies are then detected when for a test case when the prediction $\hat{y}$ of a model f deviates more than a specified threshold from the expected value y

the data at hand. For example, if the task is to detect an anomaly in an ECG, f could be a variant of an RNN, whereas if the task is to identify an anomaly in an electronic health record, then f could be a multilayer perceptron. An example application of anomaly detection in healthcare is the detection of anomalous sensor measurements such as brain waves in EEGs using DBNs [133].

4.4 Strategies for Sequential Data

Sequential data are ordered lists of events. Dealing with sequential data usually means that the input to the DL model is an ordered, variable length list of vectors. Many DL architectures and other ML methods required fixed length input. Another problem is that of the high number of input variables, as sequential data often contains many elements, which leads to neural networks with many parameters that are difficult to train.

There are three common strategies to deal with variable length input: zero padding, RNNs, and global pooling of CNNs. Zero padding means that each input sequence will be brought to equal length by adding zero vectors to lists that are short than the longest sequence. RNNs are designed to deal with sequential data of theoretically arbitrary length.

A common approach to reducing the complexity of the input data is time windowing. A windowing function w is used to summarize t time steps of input data, thereby reducing the length and consequently the complexity of the input sequences. A model is trained on the reduced data size. Figure 13 depicts the general time windowing strategy. An example of a window function is binning, i.e., multiple input events are summed up together.

More recently, 1D CNNs were used to learn the windowing function instead of handcrafting the windowing function. Similar to 2D CNNs, shared parameters are learned that describe local correlations of the input data well. CNNs can deal

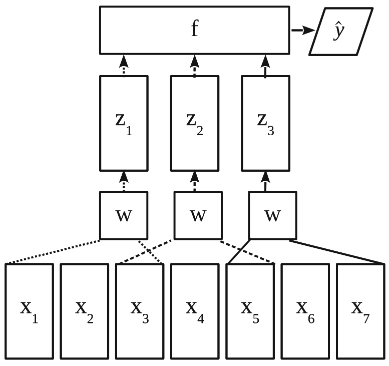

Fig. 13 Time windowing strategy for sequential data. A window function w is used to summarize a fixed number of input events and reduce the length of the sequence. A DL model is learned on the summaries z to complete a certain task

with variable length input, i.e., a long input sequence will generate a longer output sequence.

One can also combine a CNN and an RNN to analyze variable length sequential data. In this case, the CNN reduces the dimensionality of the sequential data by capturing local, spatial correlations, and the RNN learns long-term relationships between the reduced dimensionality time series. A combination of 2D CNNs and RNNs is frequently used to analyze video input.

4.5 Strategies for 2D Spatial Data

One major problem when analyzing spatial data is that the dimensions of the input data may be very high dimensional. For example, the input dimensions of a 1024×1024 image result in a prohibitively large number of neurons that need to analyze it using a fully connected neural network or other non-deep methods. One widespread and successful strategy to analyze 2D spatial is to use convolutional layers to simplify complex local correlations and the number of features that need to be processed. 2D convolutions function similarly as 1D convolution but have 2D filters instead of 1D filters, and these filters are moved in both dimensions over the input.

2D spatial data can also be considered a sequence of 2D patches. Consequentially, another strategy for analyzing 2D spatial data is to employ similar strategies as for sequential data, i.e., using RNNs or a combination of CNNs/RNNs to solve the task at hand. Treating 2D spatial data as a sequence of patches is beneficial when global contextual information is more important than local, spatial correlation. For example, Stollenga et al. proposed a multidimensional variant of an LSTM to segment tissues from images [119].

5 Deep Learning in Healthcare

DL methods are data-driven machine learning methods, which derive representations that are suitable for solving a task from data. Therefore, we describe the medical DL solutions from a data type perspective. Based on the type of signals, we will discuss suitability for different DL applications and specific methods. For each method, we will present a few solutions from the literature that are exemplary for those particular problems.

First, we will review sequential data in Sect. 5.1. Sequential data in healthcare can roughly be divided into three groups, time series, protein and DNA sequences, and longitudinal data from electronic health records. For all three of them, most tasks deal with the classification of the sequence or learning a suitable representation for the sequence so that it can be used for further analysis.

We then will provide an overview of the use of DL for analyzing spatial data in Sect. 5.2. Spatial data in healthcare are mainly images from a wide variety of devices and sources such as MRI, X-ray, or CT. The two predominant tasks are object classification and object localization. Object classification takes many forms, e.g., classifying parts of an image into cell types or classifying pixels for segmentation tasks.

Section 5.3 provides an overview of applications on health text data. Text data is mainly treated as sequential data of tokens or characters. Common DL tasks on text include sequence labeling, e.g., classification of text to identify diseases or sequence labeling for extracting named medical entities.

We did exclude a few applications and data types because they were out of scope. We will briefly mention these applications here. First, we have omitted applications of DL for analyzing spatiotemporal data such as medical videos that have both a spatial and sequential nature. Approaches generally combine methods from analyzing sequential and spatial data. Furthermore, we excluded 3D spatial data, e.g., 3D scans from an MRI. They are a more general case of the 2D spatial data, and the primary challenge is the computational complexity. Successful applications include 3D convolution and specially arranged CNN-RNN architectures that manage to reduce the computational complexity. Finally, we have excluded early DL approaches that combine handcrafted features with deep neural networks.

5.1 Applications of Deep Learning on Sequential Data

Generally, sequential data are ordered lists of events, where an event can be represented as a symbolic value, a real numerical value, a vector of real or symbolic values, or a complex data type [135].

In this chapter, we distinguish between simple sequences and time series. A simple sequence is an ordered list of symbols or an ordered list of vectors. A time series is also an ordered list of symbols or vector, but each event has a time stamp associated with it. The delta between two time stamps in a time series is commonly fixed.

Sequential data can also be considered as 1D spatial data, that is, locally correlated, and each event is referable via a 1D coordinate system. The distinguishing is arbitrary and depends on the data and the chosen data analysis techniques.

Furthermore, text can also be seen as sequential data, either as a sequence of characters or as a sequence of words. We chose to treat text data as different data modality because the relationship between the elements of text is complex.

In healthcare, there are three major sources of sequential data: electronic health records, sensor readings, and simple symbolic sequences of protein structures such as RNA, DNA, or genes. Electronic health records include a mix of demographic data and information about visits, examination results, and laboratory tests. While parts of electronic health records such as demographic data are not sequential, others such as information about each visits or longitudinal information over time can be treated sequentially. Sensor readings include time series of different medical and other sensors such as records of electrocardiogram (ECG) or electroencephalogram (EEG) signals or other bodily worn sensors. Protein structures such as DNA or RNA parts are primary sequential data sources in healthcare. In some cases, sequential data can also be represented as graph data, e.g., some DNA data is treated as functional graph.

The majority of reviewed applications of DL on sequential data in healthcare can be categorized to two groups: sequence classification and representation learning. In sequence classification, the goal is to assign one or more discrete labels to a given sequence of data, for example, to predict the sleep stage of a patient presented in an EEG input sequence. In representation learning, the primary objective is to derive features from a given input sequence that allow further data processing, e.g., classify the sequence. Other tasks that have been addressed using DL in healthcare include sequence regression, anomaly detection, and sequence denoising. In the next two sections, we will detail sequence classification and representation tasks that have been addressed using DL.

5.1.1 Representation Learning for Sequential Data

Electronic health records contain collected information on patients in a digital way. Electronic health records carry a variety of information such as medical history, laboratory test results, and demographic data. Often, this data documents a patient's development over time. The variety of data often requires to learn suitable representations to be able to analyze the data further.

Choi et al. propose Med2Vec, an approach to learn representations of electronic health records [18]. Med2Vec is based on the ideas of Doc2Vec, which is an approach for learning document representations from text data [80]. Med2Vec learns interpretable representations of sequential code- and visit-level data in an unsupervised way. Med2Vec uses a fully connected neural network to learn a representation of the sequential information of visits by solving the following task: given a task, predict the previous and the next visits. The learned embedding space allows to interpret the clinical meaning as each coordinate of the embedding vector can be viewed as a disease group. Choi et al. demonstrate the application of Med2Vec representations on the onset of heart failure prediction tasks [19], where Med2Vec yields up to 23% improvement in area under the ROC curve compared to various baselines.

Lasko et al. use autoencoders on small time windows of longitudinal serum acid measurements to discover clinical phenotypes [66]. The phenotypes are derived by Gaussian processes, which use the representations that have been extracted by the autoencoder. The derived features are as accurate as the handcrafted features.

The prediction of medical events and future diseases is another task where DL can be used to learn representations from electronic health records. Beaulieu-Jones et al. propose to use stacked denoising autoencoders to extract the phenotypes from electronic health records [8]. Denoising autoencoders are a variant of autoencoders (see Sect. 4.1), where the input is corrupted and the decoding function needs to learn to reconstruct the original input. The phenotypes that are extracted from the health records are used to predict the survival of ALS patients using random forests. Miotto et al. use denoising autoencoders in a similar fashion to predict future diseases from EHR [81]. They find that representations that are learned in such a way are useful to robustly predict future diseases such as liver cancer, diabetes, or heart failure. Their method achieves an ROC area under curve of up to 0.925.

Electrocardiogram (ECG) signals measure the electronic activity of the heart. Such data can be used to detect heart diseases and other anomalies. Huanhuan et al. propose to learn representations from ECG signals by splitting the signal to timeframes (see Sect. 4.4) and use DBNs to extract the features [48]. Using a support vector machine which uses these representations, they are able to classify heartbeats with an accuracy of 0.984. Similar to ECG data, deep neural networks can be used to learn representations from EEG data, which are used for multiple applications. Wulsin et al. use learned features to detect anomalies in EEG measurements [134]. Turner et al. use learned representations to discover whether seizures have happened in EEG data [125]. Jia et al. determine whether subjects have liked or disliked videos given features that are learned from brain waves [53]. Zhao et al. use a similar representation learning approach to detect Alzheimer's disease from EEG data [143]. Learned representations from EEG data can also be used to classify sleep stages [65].

One of the major challenges in gene expression profiling is the high dimensionality of the input sequences. Fakoor et al. propose to reduce the input dimensionality of the input space with stacked autoencoders [32]. They demonstrate that the learned representations can be used to detect various forms of cancers from gene expressions with an accuracy of up to 0.975, which is significantly better than their baseline methods. Lyons et al. and Nguyen et al. propose a similar approach to extract features from protein sequences. Lyons et al. demonstrate the applicability of the features on predicting backbone Cα angles from protein sequences [77], and Nguyen et al. use the features to assess the quality of proteins [85]. DBNs can also be used to learn representations from protein sequences. Zhang et al. use such representations to predict the RNA-binding sites of proteins [141] with an accuracy of up to 0.983. Lee et al. learn representations to predict the splicing junction [68]. Liu et al. use representations learned by a DBN from DNA sequences to identify replication domains using replication timing profiles [74]. Finally, Asgari et al. propose BioVec, ProtVec, and GenVec, three methods that are inspired by Word2Vec that learn vector representation of biological sequences, proteins, and genes [3]. They demonstrate the applicability of the learned features on classification tasks such as protein family classification, where they achieve an average classification accuracy of 93%.

5.1.2 Sequence Classification

Sequence classification is the task of assigning one or multiple discrete labels to a complete sequence. Most approaches for sequence classification follow a combination of the strategies for representation learning, classification, and sequential data (see Sects. 4.1, 4.2, 4.4). Sequence labeling is closely related to sequence classification, but instead of classifying a whole sequence, the task is to assign to parts of a sequence one or multiple labels.

For classification of sequences from electronic health records (EHRs), there are two main strategies: to use 1D CNNs or to use RNNs, both combined with a form of fully connected classification layer. Nguyen et al. predict the risk of future, unplanned readmission within 6 months after a procedure given the patients EHRs [86]. They use a 1D CNN to detect motives in the EHR and use max pooling and a Softmax layer to predict the risk. Cheng et al. propose a similar strategy to identify risks that certain diseases will develop in the future given a patient's record with an accuracy of up to 0.767 [15]. RNNs are also proposed for the tasks of predicting risks from EHR [98] as well as predicting future medication [17, 30] and predicting heart failures [20].

For clinical time series, i.e., a series of lab measurements, two strategies have been proposed. Lipton et al. and Che et al. use RNNs to predict diagnoses [73] and to predict patient mortality and ICD9 diagnosis [13]. Razavian et al. use a 1D CNN and a Softmax classifier to predict disease onset from longitudinal lab tests [100]. The DL approach significantly outperforms the baseline approaches which rely on handcrafted features.

Both CNNs and RNNs can be used to classify body and wearable signal measurements. Hammerla et al. propose a combination of convolutional and recurrent neural networks to classify activities of daily living [39]. Sathyanarayana et al. propose a method for predicting the quality of sleep states from wearable sensor data using CNNs. Their method achieves a 46% better result than the baseline approach [106]. Li et al. divide sensor data collected from RFID chips into frames and use a fully connected neural network to classify resuscitation activities [71].

A number of classification tasks on EEG and ECG data can be addressed using DL. Petrosian et al. propose to use RNNs, and Mirowski propose to use CNNs to predict seizures from EEG data [83, 97]. They achieve up to 71% sensitivity without false positives.

Stober et al. propose to use CNNs to classify EEG waves based on the rhythm that test subjects were hearing [118], achieving up to 50% per subject accuracy. Nurse et al. propose to use CNNs to predict movement controls from EEG data [90]. Finally, Pourbabaee et al. use a CNN to classify ECG data of patients to predict the risk to develop paroxysmal atrial fibrillation [99].

DNA sequences are high-dimensional data structures, and DL models allow to discover complex relationships from these structures. Similar to other classification tasks for sequences, the task that are addressed with DL on DNA and other protein sequences can be grouped into two major categories: CNNs and RNNs. Zhou et al. predict chromatin markers from DNA sequences using CNNs [144]. CNNs can also be used to predict sequence specificities of DNA- and RNA-binding proteins [1], to predict the protein binding sites [140], and to predict the cell type by DNA seq [57]. Often, these DL approaches offer a high performance, e.g., AUC over 0.92%, and in the other cases show at least a significant performance increase over the baselines.

For proteins, one of the most common tasks is secondary protein structure prediction. The classification task is to classify the amino residue to a number of discrete states, e.g., helix, sheet, or coil. Often, such secondary protein structures are predicted with a variant of an RNN. Baldi et al. propose to predict secondary protein structures with a bi-directional RNN [6] or predict such structures with a graph-based RNN [5]. Sonderby et al. propose to use an LSTM to predict secondary protein structures [115]. Instead of using an LSTM, Spencer et al. divide the protein sequences into window frames and predict the structures with a combined DBN and fully connected neural network [117]. Other tasks on proteins include the prediction of subcellular location of proteins given only the protein sequence with convolutional RNNs with an accuracy of above 0.90 [116], the detection of protein homologies using LSTMs [44], and the prediction of the protein contact map using RNNs [27].

5.1.3 Other Tasks on Sequential Data

Except for representation learning and classification of sequences, a few other tasks on sequential data can be addressed using DL. Wulsin et al. use DBNs to detect anomalies in brain waves [133]. Koh et al. use CNNs to denoise sequences to impute missing values of DNA sequences [63]. To do this, they learn a CNN, which, given a distorted input, attempts to predict the non-corrupted input. Zhu et al. use a combination of CNNs and fully connected neural networks to predict the energy expenditure of certain ambulatory activities given measurements from body sensors [145]. Their method achieves a mean regression error up to 35% lower than baseline models.

5.2 Applications of Deep Learning on Spatial Data

Depending on the content organizing data in 2D, 3D or multidimensional structures (tensors) rather than a flat vector of features (tabular data) are of significant advantage. The typical example for this is digital images where each feature corresponds to a pixel or a voxel (3D pixel). In this case the features are measurements of a sensor that is naturally organized in a spatial manner. In other words, in addition to the pixel intensities (colors), the spatial location of the information carries much of the information. Furthermore, these features demonstrate significant spatial correlation. Pixel values are highly correlated to the values of its neighbors. As an effect of this, when modeling this data, it is rather useful to look at groups of features rather than individual features. This results in spatial features that are composition of the activations of co-located finer (smaller) features. This property is commonly utilized by methods that use image data, such that features are typically edges, lines, and superpixels of similar properties. Various different techniques have been developed that detect these engineered features and enable processing of images and other data with spatial correlation. Traditional image analysis methods would build compound rule-based systems to form a hierarchical structure of features that would eventually produce the results of the analysis. DL methods also leverage the spatial properties of this data. However, rather than relying on designed rules, it uses representation learning to develop spatial features. Convolutional layers in a neural network are mainly designed for this purpose. We presented a detailed description of the CNNs in Sect. 3.3.

Images are a significant part of the spatial data of interest in the healthcare domain. To some extent, the impact of DL on healthcare is most evident on DL applications to medical image analysis. There are many sources of image data in medical and healthcare applications. A prominent area is radiology consisting of various medical imaging modalities such as MRI, CT, X-ray, and ultrasound. Furthermore, there are multiple sources of microscopy images produced in digital pathology and related domains. The number of specializations that use these modalities and their applications is broad including domains such as imaging of the brain, lung, abdomen, and retina and histopathology. Furthermore, some applications are part of the diagnostic processing, while others are in the interventional domains where the images are processed in real time.

From the perspective of image analysis, the applications are typically reduced to a set of tasks. The processing of the image can result with a label from a discrete set of labels or a continuous value. This task commonly detects an object in the image and determines the type of the image or the presence of specific patterns in the image. From a ML point of view, these tasks are referred to as classification and regression, respectively. Next, image processing can be used to detect regions of interest or assign a label to a region. This is the task of localization. In the case where each pixel of the image is assigned a class or property, the application commonly falls within the task of segmentation. Registration is the task of aligning the content of two images, and filtering is the task of processing an input image to produce an output image with specific characteristics. Even though these are some of the most common tasks, the variation and separation between them are not always clear, and some applications will fall within more than one or none of the tasks. However, this structure is useful for our purpose, since many of the DL applications within the same tasks have shared properties.

The rest of the section is organized based on the different image analysis tasks and the kind of advances that DL methods have delivered in each of them as well as the healthcare applications that rely on those technologies.

5.2.1 Classification

Classification is ML supervised learning task where the model assigns a label (class) to a data point. Medical imaging classification tasks are typically associated with a single diagnostic output variable assigned to an image. For example, in [47], Hosseini-Asl et al. present a CNN model for Alzheimer disease classification. The model processes 3D MRI images of the brain, extracts relevant features, and detects the presence of mild cognitive impairment or Alzheimer disease. Another example of image classification with DL models is presented by Shen et al. [110]. In this work, the model is used to detect lung nodules in thoracic CT images. Shen et al. demonstrate the classification of malignant and benign nodules without segmentation.

An example of this approach to digital pathology for mitotic figure count for breast cancer histology images is developed by Cirecsan et al. [23]. The model classifies patches of images that contain the cell figures as mitotic or not. This count is later used for staging and diagnosis of the disease.

Many of these models are based on the most successful DL architectures for image analysis. The introduction of the CNNs is presented in the work of LeCun et al. [67]. The breakthrough and demonstration of the capability of DL models to process images on a large scale are significantly attributed to the AlexNet model [64]. This architecture introduced a deep neural network with a large number of parameters that at the time demonstrated the best performance in assigning labels to images on the ImageNet dataset [25]. Following the initial success, some advances have been proposed in neural network architectures that have shown superior performance. However, the large number of design choices that are involved in building this model commonly makes the process complicated. Particularly for deep CNNs, the shape and size of the filters, the number of filters per layer, and the number of layers are to name a few. The VGG architecture [113] offers a simplification of facing this challenge by using a fixed size of filters (3×3). The approach captures larger features by adding multiple convolutional layers with the same filter size without subsampling layers (pooling). In this way, a set of layers acts as one layer with a wider reach and fewer parameters.

A more recent and successful development is introduced by the GoogLeNet (Inception) model [120]. This model uses an inception block consisting of multiple layers that include convolutional filters of different sizes. The module, therefore, removes the need to select the “right” size of the filters by allowing for a range of filters to work together. The inception module processes the image in multiple ways, and this combined output is provided to the next part of the model. By stacking a number of inception modules, the network has a broad capability to detect various features and build composite hierarchical representations that achieve excellent performance. One of the main strong sides of this model comes from its depth and the capability to build complex high-level features. However, even with the many advances in DL, very deep neural networks would still suffer from the vanishing gradient problem. As the depth of the model increases, the capability to train them effectively diminishes. This problem was addressed by the introduction of the skip connections in the ResNet architecture [40]. This architecture allows for effective training of very deep architectures exceeding 150 layers. The idea was then incorporated in the inception network [121]. In healthcare applications the GoogLeNet model is used in Esteva et al. [31] for the detection of skin cancer on two tasks, both discriminating between malignant and benign cases. The authors find the performance of the model in both cases on par with a dermatologist.

In the medical context, labels are expensive, and, given the possible variations of the diseases and the imaging conditions, collecting sufficient amount of annotations is a major challenge. This is also the case in the work of [31]. In this work, and many others, the authors rely on pre-training the model on another dataset and transferring this model to the medical domain where the model is fine-tuned. This technique is referred to as transfer learning. This is commonly used to deal with the lack of annotations particularly if the application warrants a large neural network with a significant amount of parameters. Transfer learning from different domains is studied by Menegola et al. [78]. Here the authors conclude that even though it would be expected that pre-training on medical datasets would be favorable to general natural images, they did not find any evidence that this has an advantage.

Another approach to deal with this problem is to use unsupervised pre-training when a significant amount of images is available without annotations. These models typically use RBMs and autoencoders to build an unsupervised representation of the data. The parameters of these models are then used to build classification models on small labeled datasets [14, 54, 126, 142].

5.2.2 Localization

Object recognition is a common task in image analysis and is frequently designed as a classification task. Detecting the presence of an object can be extended to detecting its location in different ways using similar approaches. This can be accomplished by processing patches of the image by an object detection model. In this way, the localization task is reduced to a classification task [33]. More recent approaches run bounding box proposal methods first, then followed by a classification model to achieve localization [35].

The drawback of this approach is that it is computationally inefficient given that the image needs to be processed many times by patching it, where significant parts of the image will overlap and will be processed multiple times. Furthermore, if the size of the object can vary, the patches will also need to be of different sizes, as well as the model should be able to process images with different sizes. Overall, such approaches tend to involve multiple steps of preprocessing of the data, inference, and post-processing of the model's output for successful applications.

Many of these challenges are addressed the Yolo architecture (You only look once) [101] where the whole image is processed directly. The output of this model is bounding boxes with assigned labels to them.

Application of localization with CNN to the medical domain is wide ranging. In [76], Lo et al. develop a model for localization in X-ray images.

In [24], de Vos et al. implement a model for localization of regions of interest around specific parts of the anatomy (heart, aortic arch, and descending aorta). Their method produces bounding boxes of 3D volumes by classifying 2D regions with a CNN.

Payer et al. propose a method [94] to directly regress landmark locations analogously to the approach that the Yolo model took for producing bounding boxes in a single pass. The model produces parameterized Gaussian distribution in the image space that denotes the probability of a landmark to exist effectively creating a map of landmarks on the image.

5.2.3 Semantic Segmentation

Applications that require precise annotation of each pixel are assigned to the image segmentation or semantic segmentation task. Segmentation problems typically deal with finding the boundary of objects of interest. In healthcare, this task translates to finding the boundaries of organs, cells, nuclei, blood vessels, lesions, or other objects or regions of interest. Segmentation aids either as a preprocessing step for analysis pipelines or for aiding the medical professional. In this context semantic segmentation determines which pixels belong to a particular organ or tissues. Extracting accurate boundaries can be challenging for multiple reasons: the input data may be noisy, the input data often lacks contrasting edges, and the objects of interest may vary significantly mainly due to specific pathologies.

To achieve semantic segmentation with deep neural networks, the task can be reduced to classification. In this case, the model sees a patch of the image and assigns a label to a specific pixel. Pixel-wise classification is applied to skin lesion detection [56]. The drawback of such methods is that it is computational complex because the overlapping patches results in processing images many times. To improve on this approach, the model would need to handle the whole image in a single pass and produce a segmentation map for the whole image.

Typically CNNs process the image in stages, and for efficiency large amount of information is removed in each step. This works very well for classification tasks; however, for localization, the output is very detailed since we assign a class for each pixel. In this context, highly detailed information is important to achieve pixel-level accuracy. At the same time, global information from different parts of the image is also important to bring the context for each of the decisions. The UNet [103] architecture does this very well. It processes the information in steps reducing the details and producing global context. However, it also introduces "skip" connections that can carry the locally important details at each level such that the final segmentation can be achieved by combining both global context and local details.

This work is extended for 3D segmentations by Cicek et al. in [22]. Another application of a network architecture similar to U-Net that uses both the global and local contexts to assign semantics to the pixels is presented by Brosch et al. in [24]. Brosch et al. present a model that segments white matter lesions in brain MRI.

5.2.4 Registration

Registration is the task of spatially aligning data about an object from different sources. The data which should be combined may come from various devices and be recorded from different angles or with a different technology.

DL's main contribution to registration is derived from CNNs' capability to represent locally correlated, spatial data well. Methods are mainly based on successful architectures from other domains such as VGGNet [113] or U-Net [103]. Representations learned by these architectures are used to calculate a similarity score to determine overlapping areas.

Wu et al. use a stacked convolutional autoencoder to extract essential features from brain MRI images. To determine whether two of these representation patches are overlapping, they calculate the normalized cross-correlation. Their method achieves an overall 2.74% improvement [131].

Instead of learning representations from the images, Yang et al. propose to use a convolutional encoder-decoder network to learn to predict the pixel-wise momentum-parameterization. The encoder and decoding networks resemble a VGGNet [137], and they test their approach on the OASIS longitudinal dataset. Their approach is released as freely available software [139].

Miao et al. frame the registration problem as regression task [79]. They train a CNN to learn a mapping between a 3D X-ray and a digitally reconstructed radiograph by estimating the difference of their underlying transformation parameters.

Simonovsky et al. frame the registration problem as classification task [112]. They train a two-channel, five-layer CNN to discriminate between aligned and misaligned patches from different modalities to align neonatal brain images.

5.2.5 Quality Enhancement

Medical image quality enhancement methods are based on the advancement of the image processing domain. Image quality enhancement is mainly concerned with three different tasks: denoising input images, imputing missing data, and increasing the resolution of low-resolution images.

The fundamental idea of image denoising using deep neural networks is that the problem can be described as a mapping from an image with noisy to a noise-free image [12]. To learn such a function, noise is added to an image, and given a noisy image, the noise-free version needs to be constructed. The noise can be domain-specific such as dust particles on the image, or it can be domain independent, e.g., Gaussian noise. Such a denoising function can be learned in an unsupervised way. More generally, denoising autoencoders can be used to learn robust representations from input data [127]. Benaou et al. propose to use an ensemble of stacked, denoising autoencoders to improve the contrast from MRI images [11]. Each member of the ensemble is trained with a different noise type, and the result is selected via a Softmax activation function. Janowczyk et al. apply a similar strategy to normalize the colors of H&E stained histopathology images [52].

Instead of an ensemble of different autoencoders, they normalize the images by color deconvolution followed by thresholding of the images.

Another application for enhancing images is proposed by Yang et al. [138]. They use CNNs to suppress bone structures in chest X-rays. The CNN learns a mapping between chest X-rays and their bone components in the gradient domain. They base their approach on the edge filtering architecture that is proposed in [136], which resembles a denoising autoencoder. Instead of adding noise to the image, they apply edge detection filters.

Image super-resolution is the task of deriving a high-resolution from a low-resolution image. Dong et al. propose a DL method based on CNNs to tackle this task [28]. Their approach is structured like a CNN autoencoder, with one CNN that learns to represent the input patches and a second one that learns to construct high-resolution images from this representation. In healthcare, Oktay et al. improve the layer design and the training procedure of this strategy to enhance the resolution of cardiac MRIs [91].

Also using CNN autoencoders, Nie et al. derive CT images from MRI images [88], and Bahrami et al. derive high-quality 7T MRI images from lower-quality 3T MRI images [4].

Deep neural networks can also be used to impute missing data. Conceptually, the task of imputing missing data from images is very similar to denoising them. But, instead of adding noise to the input images, parts of the input image will be removed, and the task of the network is to learn to reconstruct the complete image given the incomplete one [92]. Within healthcare, Li et al. use 3D CNNs to learn to reconstruct missing PET patterns from MRI images [70].

5.3 Applications of Deep Learning on Text Data

Text data is a form of unstructured, sequential data. Commonly, a text is interpreted in two ways: as sequence of characters and as sequence of tokens where tokens can be words, word parts, punctuation, or stopping characters. The elements of a time series are generally related to each other in a linear, temporal fashion, i.e., elements from earlier time steps. This relationship is often also true for text data. However, the relationships between the elements are more complicated, often long-term, and high-level.

In the healthcare domain, text data occurs mostly in electronic health records (EHRs) in the form of text messages, clinical notes, medical notes, memos, or free text entries in medical databases. More recently, few works have analyzed health-related issues using social media data, e.g., predicting the outbreaks of infectious diseases using text from social media [146].

Text data is generally treated as sequential data and analyzed with similar means. One problem of analyzing text is to find suitable representations of the input words. Text data is high dimensional, which renders sparse representations computationally impractical. Hence, commonly dense representations such as Word2Vec [80] and

GloVe [96] are used to represent input words. In healthcare, analyzing text centers around three major problems: question answering, information extraction, and finding a suitable representation of medical text for further analysis.

5.3.1 Question Answering by Sequence Classification

Nie et al. address a medical question answering problem by framing it as a sequence classification problem [87]. They use a sparsely connected, multilayer neural network that is pre-trained in an unsupervised way to infer soft layers of fixed frames of the question text. Their method allows inferring diseases of patients given a free text question that these patients ask with an accuracy of up to 98.21%. Jacobsen et al. propose a method for disease prediction from electronic health records. This method also treats text as a sequence of tokens [49]. Their best-performing model is a combination of a deep belief network and a fully connected classification layer, which yields an $F1$ score of 0.81.

Tweets are short text messages that do not contain medical data per se, but they can be used to analyze specific general trends in population health. Kendra et al. treat tweets as sequence of tokens to classify them into medical categories by using a fully connected neural network [58]. Zou et al. use the text data from tweets to predict the outbreaks of infectious diseases [146]. They combine Word2Vec skip-gram [80] representations of the tweets with elastic nets to predict the time of outbreak of a specific disease, e.g., Norovirus of food poisoning.

Dernoncourt et al. concern themselves with the privacy of patients. They show that it is possible to de-identify patients from text in electronic health records by framing the task as a sequence classification problem. They use a bi-directional LSTM to encode text from EHRs to predict a patient's ID. Their best-performing model achieves an $F1$ score of 99.23 [26].

5.4 Information Extraction by Sequence Labeling

Jagannatha et al. propose a DL-based method to extract medical entities like medication from EHR text. To do this, they frame the problem as a sequence-labeling problem comparable to named entity recognition problems from the natural language processing domain. Their first proposed method evaluates various forms of bi-directional RNNs on their capability of labeling medical entities from electronic health records [50]. Their best-performing model achieves a labeling $F1$ score of 0.813. They extend their method by combining the bi-directional RNNs with conditional random fields and achieve an $F1$ score of 0.8614 on the same task [51]. Wu et al. address a similar problem using a combination of context vectors, CNNs, and a fully connected classification layer to recognize medical events [132]. Their method achieves an $F1$ score of 0.928.

5.4.1 Representation Learning of Medical Text

Tran et al. propose to learn suitable representations from a text of electronic health records using RBMs [124]. They demonstrate the applicability of representations that have been learned by risk group discovery. Depending on the group, they achieve an $F1$ score of 0.359, which is roughly a 5.6% performance improvement over the baseline.

Liu et al. propose to learn word embeddings to expand medical abbreviations [75]. Their methods calculate the similarity score between the learned representations of large, medical text corpora. Their best-performing model achieves an accuracy of 82.27%, which is better than the accuracy of a general physician (80%) but worse than the accuracy of a domain expert that has received additional training in the respective field (>90%).

6 Conclusion

Deep Learning methods learn a hierarchy of representations from data. This multi-layered architecture allows solving machine learning problems more efficiently than shallow machine learning methods. Furthermore, it reduces the need for manually designing data representations, because each layer automatically learns representations that are useful for next layers to solve a task. Learning representations has a few advantages over manually designing the features. The most striking of them is solving a task which becomes independent of the domain of the data. For example, Deep Learning methods that address face recognition tasks can now easily be used for a medical segmentation tasks, since the underlying data—spatial data—is similar in structure. In contrast to that, handcrafted features for face recognition are challenging to use for a medical segmentation task. This domain independence of Deep Learning methods combined with generally good performance on many tasks has led to the adoption of Deep Learning in many domains and also in healthcare.

In this chapter, we presented advances in Deep Learning in healthcare from a data analysis perspective. Machine learning is already widely used in the healthcare domain for a variety of tasks, and recently Deep Learning has become more prominent. We motivated the use of Deep Learning over “traditional” machine learning by highlighting the significant problems that Deep Learning addresses: it provides an efficient way to learn distributed representations which are domain independent; it also provides an effective strategy for addressing the curse of dimensionality.

Deep Learning methods have shown remarkable success on many healthcare-related tasks and also on tasks of many other domains. There are, however, a large number of open challenges in Deep Learning that are hardly addressed up to now but need to be solved.

While Deep Learning methods perform remarkably well on many tasks, their inner workings still remain poorly understood. In many domains, but particularly in healthcare, being able to justify and explain a prediction made by a machine learning model is very important for two reasons. First, if a model can justifiably tell why a specific prediction was made, it will increase the trust of a user in the prediction, and secondly, it will provide means to validate the prediction. In other domains, first steps toward interpretable predictions have been made, e.g., [55, 69, 102], but interpretability of Deep Learning model predictions still remains an unsolved challenge.

Deep Learning learns hierarchies of representations from data to solve specific tasks. Other issues that are rarely addressed but are essential for practical use in healthcare are methods to ensure that the data is (a) representative for the tasks that are being addressed and (b) free from biases. The models that are trained with Deep Learning can only be as good as the data that was used to learn them. Any conclusions about the validity of a decision or prediction depend on the confidence that the data that was learned was bias-free and representative. Generally, if a dataset is large enough, it is assumed to be normal. In healthcare, datasets are often small and may not be representative. Methods to ensure the quality of the data would be required in this case.

Another challenge that needs to be addressed is to ensure consistency of performance on the deployment in real-life systems. Many academic DL methods are currently trained on benchmark datasets, which may or may not be representative of data in real-life applications. More research would be needed to guarantee robustness of these methods.

Finally, another regulatory approval for DL methods is a challenge that is currently poorly addressed but crucial for the healthcare domain. Progress on the challenges of justifiable predictions, data quality assurance, and robustness will be required to make Deep Learning methods suitable for regulatory approval by organizations such as the Food and Drug Administration in the United States, the European Medicines Agency in the EU, or the China Food and Drug Administration.

References

1. Alipanahi, B., Delong, A., Weirauch, M.T., Frey, B.J.: Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. Nat. Biotechnol. **33**(8), 831–838 (2015). https://doi.org/10.1038/nbt.3300.
2. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein gan (2017). arXiv preprint arXiv:1701.07875
3. Asgari, E., Mofrad, M.R.K.: Continuous distributed representation of biological sequences for deep proteomics and genomics. PLoS One **10**(11), e0141287 (2015)
4. Bahrami, K., Shi, F., Rekik, I., Shen, D.: Convolutional neural network for reconstruction of 7T-like images from 3T MRI using appearance and anatomical features. In: Deep Learning and Data Labeling for Medical Applications, pp. 39–47. Springer, New York (2016)
5. Baldi, P., Pollastri, G.: The principled design of large-scale recursive neural network architectures–DAG-RNNs and the protein structure prediction problem. J. Mach. Learn. Res. **4**, 575–602 (2003)

6. Baldi, P., Brunak, S., Frasconi, P., Soda, G., Pollastri, G.: Exploiting the past and the future in protein secondary structure prediction. Bioinformatics **15**(11), 937–946 (1999)
7. Banks, G.: Artificial intelligence in medical diagnosis: the INTERNIST/CADUCEUS approach. Crit. Rev. Med. Inf. **1**(1), 23–54 (1986)
8. Beaulieu-Jones, B.K., Greene, C.S., et al.: Semi-supervised learning of the electronic health record for phenotype stratification. J. Biomed. Inf. **64**, 168–178 (2016)
9. Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H.: Greedy layer-wise training of deep networks. In: Advances in Neural Information Processing Systems, pp. 153–160 (2007)
10. Bengio, Y., Courville, A., Vincent, P.: Representation learning: a review and new perspectives. IEEE Trans. Pattern Anal. Mach. Intell. **35**(8), 1798–1828 (2013). https://doi.org/10.1109/TPAMI.2013.50
11. Benou, A., Veksler, R., Friedman, A., Raviv, T.R.: De-noising of contrast-enhanced MRI sequences by an ensemble of expert deep neural networks. In: Deep Learning and Data Labeling for Medical Applications, pp. 95–110. Springer, New York (2016)
12. Burger, H.C., Schuler, C.J., Harmeling, S.: Image denoising: can plain neural networks compete with BM3D? In: 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2392–2399. IEEE, New York (2012)
13. Che, Z., Purushotham, S., Cho, K., Sontag, D., Liu, Y.: Recurrent neural networks for multivariate time series with missing values. Sci. Rep. **8**(1), 6085 (2018)
14. Cheng, J.Z., Ni, D., Chou, Y.H., Qin, J., Tiu, C.M., Chang, Y.C., Huang, C.S., Shen, D., Chen, C.M.: Computer-aided diagnosis with deep learning architecture: applications to breast lesions in US images and pulmonary nodules in CT scans. Sci. Rep. **6**, 24454 (2016)
15. Cheng, Y., Wang, F., Zhang, P., Hu, J.: Risk prediction with electronic health records: a deep learning approach. In: Proceedings of the 2016 SIAM International Conference on Data Mining, pp. 432–440. SIAM, Philadelphia (2016)
16. Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1724–1734 (2014). https://doi.org/10.3115/v1/D14-1179; http://arxiv.org/abs/1406.1078
17. Choi, E., Bahadori, M.T., Schuetz, A., Stewart, W.F., Sun, J.: Doctor AI: predicting clinical events via recurrent neural networks. In: Machine Learning for Healthcare Conference, pp. 301–318 (2016)
18. Choi, E., Bahadori, M.T., Searles, E., Coffey, C., Thompson, M., Bost, J., Tejedor-Sojo, J., Sun, J.: Multi-layer representation learning for medical concepts. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1495–1504. ACM, New York (2016)
19. Choi, E., Schuetz, A., Stewart, W.F., Sun, J.: Medical concept representation learning from electronic health records and its application on heart failure prediction (2016). arXiv preprint arXiv:1602.03686
20. Choi, E., Schuetz, A., Stewart, W.F., Sun, J.: Using recurrent neural network models for early detection of heart failure onset. J. Am. Med. Inf. Assoc. **24**(2), 361–370 (2016)
21. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 539–546 (2005). https://doi.org/10.1109/CVPR.2005.202
22. Çiçek, z., Abdulkadir, A., Lienkamp, S.S., Brox, T., Ronneberger, O.: 3D U-Net: learning dense volumetric segmentation from sparse annotation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 424–432. Springer, New York (2016)
23. Cireşan, D.C., Giusti, A., Gambardella, L.M., Schmidhuber, J.: Mitosis detection in breast cancer histology images with deep neural networks. In: International Conference on Medical Image Computing and Computer-assisted Intervention, pp. 411–418. Springer, New York (2013)

24. de Vos, B.D., Wolterink, J.M., de Jong, P.A., Viergever, M.A., Išgum, I.: 2D image classification for 3D anatomy localization: employing deep convolutional neural networks. In: International Society for Optics and Photonics (2016), 97841Y. https://doi.org/10.1117/12.2216971; http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.2216971
25. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: a large-scale hierarchical image database. In: IEEE Conference on Computer Vision and Pattern Recognition, 2009, CVPR 2009, pp. 248–255. IEEE, New York (2009)
26. Dernoncourt, F., Lee, J.Y., Uzuner, O., Szolovits, P.: De-identification of patient notes with recurrent neural networks. J. Am. Med. Inf. Assoc. **24**(3), 596–606 (2017)
27. Di Lena, P., Nagata, K., Baldi, P.: Deep architectures for protein contact map prediction. Bioinformatics **28**(19), 2449–2457 (2012)
28. Dong, C., Loy, C.C., He, K., Tang, X.: Image super-resolution using deep convolutional networks. IEEE Trans. Pattern Anal. Mach. Intell. **38**(2), 295–307 (2016). https://doi.org/10.1109/TPAMI.2015.2439281
29. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. J. Mach. Learn. Res. **12**, 2121–2159 (2011)
30. Esteban, C., Staeck, O., Baier, S., Yang, Y., Tresp, V.: Predicting clinical events by combining static and dynamic information using recurrent neural networks. In: 2016 IEEE International Conference on Healthcare Informatics (ICHI), pp. 93–101. IEEE, New York (2016)
31. Esteva, A., Kuprel, B., Novoa, R.A., Ko, J., Swetter, S.M., Blau, H.M., Thrun, S.: Dermatologist-level classification of skin cancer with deep neural networks. Nature **542**(7639), 115 (2017)
32. Fakoor, R., Ladhak, F., Nazi, A., Huber, M.: Using deep learning to enhance cancer diagnosis and classification. In: Proceedings of the International Conference on Machine Learning, vol. 28 (2013)
33. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE Trans. Pattern Anal. Mach. Intell. **32**(9), 1627–1645 (2010). https://doi.org/10.1109/TPAMI.2009.167. http://ieeexplore.ieee.org/document/5255236/
34. Geman, S., Doursat, R., Bienenstock, E.: Neural networks and the bias/variance dilemma. Neural Comput. **4**(1), 1–58 (1992). https://doi.org/10.1162/neco.1992.4.1.1
35. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation (2013). http://arxiv.org/abs/1311.2524
36. Goodfellow, I.J.: On distinguishability criteria for estimating generative models (2014). arXiv preprint arXiv:1412.6515
37. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in Neural Information Processing Systems, pp. 2672–2680 (2014)
38. Graves, A., Wayne, G., Danihelka, I.: Neural turing machines. 1–26 (2014). arXiv. https://doi.org/10.3389/neuro.12.006.2007. http://arxiv.org/abs/1410.5401
39. Hammerla, N.Y., Halloran, S., Plötz, T.: Deep, convolutional, and recurrent models for human activity recognition using wearables. In: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, pp. 1533–1540. AAAI Press, Palo Alto (2016)
40. 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, pp. 770–778 (2016)
41. Hinton, G.E., McClelland, J.L., Rumelhart, D.E.: Distributed representations. In: Rumelhart, D.E., McClelland, J.L., CORPORATE PDP Research Group (eds.) Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol. 1. MIT Press, Cambridge (1986)
42. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. Neural Comput. **18**(7), 1527–1554 (2006)
43. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)

44. Hochreiter, S., Heusel, M., Obermayer, K.: Fast model-based protein homology detection without alignment. Bioinformatics **23**(14), 1728–1736 (2007)
45. Hoffer, E., Ailon, N.: Deep metric learning using triplet network. In: International Workshop on Similarity-Based Pattern Recognition, pp. 84–92. Springer, New York (2015)
46. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Netw. **2**(5), 359–366 (1989)
47. Hosseini-Asl, E., Gimel'farb, G., El-Baz, A.: Alzheimer's disease diagnostics by a deeply supervised adaptable 3D convolutional network (2016). arXiv preprint arXiv:1607.00556
48. Huanhuan, M., Yue, Z.: Classification of electrocardiogram signals with deep belief networks. In: 2014 IEEE 17th International Conference on Computational Science and Engineering (CSE), pp. 7–12. IEEE, New York (2014)
49. Jacobson, O., Dalianis, H.: Applying deep learning on electronic health records in Swedish to predict healthcare-associated infections. In: Proceedings of the 15th Workshop on Biomedical Natural Language Processing, pp. 191–195 (2016)
50. Jagannatha, A.N., Yu, H.: Bidirectional RNN for medical event detection in electronic health records. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics Meeting, vol. 2016, p. 473. NIH Public Access (2016)
51. Jagannatha, A.N., Yu, H.: Structured prediction models for RNN based sequence labeling in clinical text. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, vol. 2016, p. 856. NIH Public Access (2016)
52. Janowczyk, A., Basavanhally, A., Madabhushi, A.: Stain normalization using sparse autoencoders (StaNoSA): application to digital pathology. Comput. Med. Imag. Graph. **57**, 50–61 (2017)
53. Jia, X., Li, K., Li, X., Zhang, A.: A novel semi-supervised deep learning framework for affective state recognition on EEG signals. In: 2014 IEEE International Conference on Bioinformatics and Bioengineering (BIBE), pp. 30–37. IEEE, New York (2014)
54. Kallenberg, M., Petersen, K., Nielsen, M., Ng, A.Y., Diao, P., Igel, C., Vachon, C.M., Holland, K., Winkel, R.R., Karssemeijer, N., et al.: Unsupervised deep learning applied to breast density segmentation and mammographic risk scoring. IEEE Trans. Med. Imag. **35**(5), 1322–1331 (2016)
55. Karpathy, A., Johnson, J., Fei-Fei, L.: Visualizing and understanding recurrent networks. In: ICLR, pp. 1–13 (2016). https://doi.org/10.1007/978-3-319-10590-1_53
56. Kawahara, J., Brown, C.J., Miller, S.P., Booth, B.G., Chau, V., Grunau, R.E., Zwicker, J.G., Hamarneh, G.: BrainNetCNN: convolutional neural networks for brain networks; towards predicting neurodevelopment. NeuroImage **146**, 1038–1049 (2017)
57. Kelley, D.R., Snoek, J., Rinn, J.L.: Basset: learning the regulatory code of the accessible genome with deep convolutional neural networks. Genome Res. **26**(7), 990–999 (2016)
58. Kendra, R.L., Karki, S., Eickholt, J.L., Gandy, L.: Characterizing the discussion of antibiotics in the twittersphere: what is the bigger picture? J. Med. Internet Res. **17**(6), e154 (2015)
59. Keogh, E., Mueen, A.: Curse of dimensionality. In: Encyclopedia of Machine Learning, pp. 257–258. Springer, New York (2011)
60. Khademi, M., Nedialkov, N.S.: Probabilistic graphical models and deep belief networks for prognosis of breast cancer. In: 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA), pp. 727–732. IEEE, New York (2015)
61. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization (2014). http://arxiv.org/abs/1412.6980
62. Kingma, D.P., Welling, M.: Auto-encoding variational Bayes (2013). arXiv preprint arXiv:1312.6114
63. Koh, P.W., Pierson, E., Kundaje, A.: Denoising genome-wide histone ChIP-seq with convolutional neural networks. Bioinformatics (Oxford, England) **33**(14), i225–i233 (2017)
64. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (eds.) Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1 (NIPS'12), pp. 1097–1105. Curran Associates Inc., Red Hock (2012)

65. Längkvist, M., Karlsson, L., Loutfi, A.: Sleep stage classification using unsupervised feature learning. Adv. Artif. Neural Syst. **2012**, 9 (2012)
66. Lasko, T.A., Denny, J.C., Levy, M.A.: Computational phenotype discovery using unsupervised feature learning over noisy, sparse, and irregular clinical data. PLoS One **8**(6), e66341 (2013)
67. LeCun, Y., Jackel, L., Cortes, C.: Learning algorithms for classification: a comparison on handwritten digit recognition. https://pdfs.semanticscholar.org/943d/6db0c56a5f4d04a3f81db633fec7cc4fde0f.pdf
68. Lee, T., Yoon, S.: Boosted categorical restricted Boltzmann machine for computational prediction of splice junctions. In: International Conference on Machine Learning, pp. 2483–2492 (2015)
69. Lei, T., Barzilay, R., Jaakkola, T.: Rationalizing neural predictions. In: EMNLP 2016, Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 107–117 (2016). http://arxiv.org/abs/1606.04155
70. Li, R., Zhang, W., Suk, H.I., Wang, L., Li, J., Shen, D., Ji, S.: Deep learning based imaging data completion for improved brain disease diagnosis. In: International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 305–312. Springer, New York (2014)
71. Li, X., Zhang, Y., Li, M., Marsic, I., Yang, J., Burd, R.S.: Deep neural network for RFID-based activity recognition. In: Pour, Y.G. (ed.) S3@MobiCom, pp. 24–26. ACM, New York (2016)
72. Liao, R., Miao, S., de Tournemire, P., Grbic, S., Kamen, A., Mansi, T., Comaniciu, D.: An artificial agent for robust image registration. In: Proceedings of the Thirty-First {AAAI} Conference on Artificial Intelligence, February 4–9, 2017, San Francisco, CA, pp. 4168–4175 (2017). http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14751
73. Lipton, Z.C., Kale, D.C., Elkan, C., Wetzel, R.: Learning to diagnose with LSTM recurrent neural networks (2015). arXiv preprint arXiv:1511.03677
74. Liu, F., Ren, C., Li, H., Zhou, P., Bo, X., Shu, W.: De novo identification of replication-timing domains in the human genome by deep learning. Bioinformatics **32**(5), 641–649 (2015)
75. Liu, Y., Ge, T., Mathews, K.S., Ji, H., McGuinness, D.L.: Exploiting task-oriented resources to learn word embeddings for clinical abbreviation expansion (2018). arXiv preprint arXiv:1804.04225
76. Lo, S.C., Lou, S.L., Lin, J.S., Freedman, M.T., Chien, M.V., Mun, S.K.: Artificial convolution neural network techniques and applications for lung nodule detection. IEEE Trans. Med. Imag. **14**(4), 711–718 (1995)
77. Lyons, J., Dehzangi, A., Heffernan, R., Sharma, A., Paliwal, K., Sattar, A., Zhou, Y., Yang, Y.: Predicting backbone C\$$\alpha$\$ angles and dihedrals from protein sequences by stacked sparse auto-encoder deep neural network. J. Comput. Chem. **35**(28), 2040–2046 (2014)
78. Menegola, A., Fornaciali, M., Pires, R., Avila, S., Valle, E.: Towards automated melanoma screening: exploring transfer learning schemes (2016). arXiv preprint arXiv:1609.01228
79. Miao, S., Wang, Z.J., Liao, R.: A CNN regression approach for real-time 2D/3D registration. IEEE Trans. Med. Imag. **35**(5), 1352–1363 (2016)
80. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Burges, C.J.C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 26, pp. 3111–3119. Curran Associates, Inc., Red Hook (2013)
81. Miotto, R., Li, L., Kidd, B.A., Dudley, J.T.: Deep patient: an unsupervised representation to predict the future of patients from the electronic health records. Sci. Rep. **6**, 26094 (2016)
82. Miotto, R., Wang, F., Wang, S., Jiang, X., Dudley, J.T.: Deep learning for healthcare: review, opportunities and challenges. Brief. Bioinform. **19**(6), 1236–1246 (2018)
83. Mirowski, P., Madhavan, D., LeCun, Y., Kuzniecky, R.: Classification of patterns of EEG synchronization for seizure prediction. Clin. Neurophysiol. **120**(11), 1927–1940 (2009)
84. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10), pp. 807–814 (2010)

85. Nguyen, S.P., Shang, Y., Xu, D.: DL-PRO: a novel deep learning method for protein model quality assessment. In: 2014 International Joint Conference on Neural Networks (IJCNN), pp. 2071–2078. IEEE, New York (2014)
86. Nguyen, P., Tran, T., Wickramasinghe, N., Venkatesh, S.: Deepr: a convolutional net for medical records. IEEE J. Biomed. Health Inf. **21**(1), 22–30 (2017)
87. Nie, L., Wang, M., Zhang, L., Yan, S., Zhang, B., Chua, T.S.: Disease inference from health-related questions via sparse deep learning. IEEE Trans. Knowl. Data Eng. **27**(8), 2107–2119 (2015)
88. Nie, D., Cao, X., Gao, Y., Wang, L., Shen, D.: Estimating CT image from MRI data using 3D fully convolutional networks. In: Deep Learning and Data Labeling for Medical Applications, pp. 170–178. Springer, New York (2016)
89. Nie, D., Trullo, R., Lian, J., Petitjean, C., Ruan, S., Wang, Q., Shen, D.: Medical image synthesis with context-aware generative adversarial networks. In: International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 417–425. Springer, New York (2017)
90. Nurse, E., Mashford, B.S., Yepes, A.J., Kiral-Kornek, I., Harrer, S., Freestone, D.R.: Decoding EEG and LFP signals using deep learning: heading TrueNorth. In: Proceedings of the ACM International Conference on Computing Frontiers, pp. 259–266. ACM, New York (2016)
91. Oktay, O., Bai, W., Lee, M., Guerrero, R., Kamnitsas, K., Caballero, J., de Marvao, A., Cook, S., O'Regan, D., Rueckert, D.: Multi-input cardiac image super-resolution using convolutional neural networks. In: International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 246–254. Springer, New York (2016)
92. Oord, A.V.D., Kalchbrenner, N., Kavukcuoglu, K.: Pixel recurrent neural networks. In: International Conference on Machine Learning (ICML) (2016). http://arxiv.org/abs/1601.06759
93. Pascanu, R., Mikolov, T., Bengio, Y.: On the difficulty of training recurrent neural networks. In: ICML (3), vol. 28, pp. 1310–1318 (2013)
94. Payer, C., Štern, D., Bischof, H., Urschler, M.: Regressing heatmaps for multiple landmark localization using CNNs. pp. 230–238. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46723-8_27.
95. Peng, C.Y.J., Lee, K.L., Ingersoll, G.M.: An introduction to logistic regression analysis and reporting. J. Educ. Res. **96**(1), 3–14 (2002)
96. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: EMNLP, vol. 14, pp. 1532–1543 (2014)
97. Petrosian, A., Prokhorov, D., Homan, R., Dasheiff, R., Wunsch II, D.: Recurrent neural network based prediction of epileptic seizures in intra-and extracranial EEG. Neurocomputing **30**(1–4), 201–218 (2000)
98. Pham, T., Tran, T., Phung, D., Venkatesh, S.: Deepcare: a deep dynamic memory model for predictive medicine. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 30–41. Springer, Berlin (2016)
99. Pourbabaee, B., Roshtkhari, M.J., Khorasani, K.: Deep convolutional neural networks and learning ECG features for screening paroxysmal atrial fibrillation patients. IEEE Trans. Syst. Man Cybernet. Syst. **48**(12), 2095–2104 (2017)
100. Razavian, N., Marcus, J., Sontag, D.: Multi-task prediction of disease onsets from longitudinal lab tests (2016). arXiv preprint arXiv:1608.00647
101. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection (2015). http://arxiv.org/abs/1506.02640
102. Ribeiro, M.T., Singh, S., Guestrin, C.: " Why should I trust you?": explaining the predictions of any classifier (2016). arXiv preprint arXiv:1602.04938
103. 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, pp. 234–241. Springer, Cham (2015)
104. Sabour, S., Frosst, N., Hinton, G.E.: Dynamic routing between capsules. In: Advances in Neural Information Processing Systems, pp. 3859–3869 (2017)

105. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training GANs. In: NIPS, pp. 1–10 (2016). arXiv:1504.01391
106. Sathyanarayana, A., Joty, S., Fernandez-Luque, L., Ofli, F., Srivastava, J., Elmagarmid, A., Arora, T., Taheri, S.: Sleep quality prediction from wearable data using deep learning. JMIR mHealth and uHealth **4**(4), e130 (2016)
107. Schmidhuber, J.: Deep learning in neural networks: an overview. Neural Netw. **61**, 85–117 (2015). https://doi.org/10.1016/j.neunet.2014.09.003. http://arxiv.org/abs/1404.7828
108. Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., Dean, J.: Outrageously large neural networks: the sparsely-gated mixture-of-experts layer (2017). arXiv preprint arXiv:1701.06538
109. Shen, W., Zhou, M., Yang, F., Yang, C., Tian, J.: Multi-scale convolutional neural networks for lung nodule classification. In: International Conference on Information Processing in Medical Imaging, pp. 588–599. Springer, Cham (2015)
110. Shen, W., Zhou, M., Yang, F., Dong, D., Yang, C., Zang, Y., Tian, J.: Learning from Experts: Developing Transferable Deep Features for Patient-Level Lung Cancer Prediction. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 9901, pp. 124–131 (2016). https://doi.org/10.1007/978-3-319-46723-8_15. https://www.scopus.com/inward/record.uri?eid=2-s2.0-84996497545&doi=10.1007%2F978-3-319-46723-8_15&partnerID=40&md5=e5253c871ee40426de6895cf297af84b
111. Shwartz-Ziv, R., Tishby, N.: Opening the Black Box of Deep Neural Networks via Information. CoRR:abs/1703.0 (2017). http://arxiv.org/abs/1703.00810
112. Simonovsky, M., Gutiérrez-Becker, B., Mateus, D., Navab, N., Komodakis, N.: A deep metric for multimodal registration. In: International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 10–18. Springer, Basel (2016)
113. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2014). ArXiv e-prints
114. Smolensky, P.: Information processing in dynamical systems: foundations of harmony theory. Technical report, Colorado Univ at Boulder Dept of Computer Science (1986)
115. Sønderby, S.K., Winther, O.: Protein secondary structure prediction with long short term memory networks (2014). arXiv preprint arXiv:1412.7828
116. Sønderby, S.K., Sønderby, C.K., Nielsen, H., Winther, O.: Convolutional LSTM networks for subcellular localization of proteins. In: International Conference on Algorithms for Computational Biology, pp. 68–80. Springer, Heidelberg (2015)
117. Spencer, M., Eickholt, J., Cheng, J.: A deep learning network approach to ab initio protein secondary structure prediction. IEEE/ACM Trans. Comput. Biol. Bioinform. **12**(1), 103–112 (2015)
118. Stober, S., Cameron, D.J., Grahn, J.A.: Using convolutional neural networks to recognize rhythm stimuli from electroencephalography recordings. In: Advances in Neural Information Processing Systems, pp. 1449–1457 (2014)
119. Stollenga, M.F., Byeon, W., Liwicki, M., Schmidhuber, J.: Parallel multi-dimensional LSTM, with application to fast biomedical volumetric image segmentation. In: Advances in Neural Information Processing Systems, pp. 2998–3006 (2015)
120. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 07–12 June, pp. 1–9 (2015). https://doi.org/10.1109/CVPR.2015.7298594
121. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.: Inception-v4, inception-resnet and the impact of residual connections on learning (2017). http://www.aaai.org/ocs/index.php/AAAI/AAAI17/paper/download/14806/14311
122. Taylor, K.: Connected Health: How Digital Technology is Transforming Health and Social Care. Deloitte Centre for Health Solutions, London (2015)
123. Tieleman, T., Hinton, G.: Lecture 6.5-rmsprop: divide the gradient by a running average of its recent magnitude. COURSERA: Neural Netw. Mach. Learn. **4**(2), 26–31 (2012)

124. Tran, T., Nguyen, T.D., Phung, D., Venkatesh, S.: Learning vector representation of medical objects via EMR-driven nonnegative restricted Boltzmann machines (eNRBM). J. Biomed. Inf. **54**, 96–105 (2015)
125. Turner, J.T., Page, A., Mohsenin, T., Oates, T.: Deep belief networks used on high resolution multichannel electroencephalography data for seizure detection. In: 2014 AAAI Spring Symposium Series (2014)
126. van Tulder, G., de Bruijne, M.: Combining generative and discriminative representation learning for lung CT analysis with convolutional restricted Boltzmann machines. IEEE Trans. Med. Imag. **35**(5), 1262–1272 (2016)
127. Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A.: Extracting and composing robust features with denoising autoencoders. In: Proceedings of the 25th International Conference on Machine Learning - ICML '08, pp. 1096–1103. ACM, New York (2008). https://doi.org/10.1145/1390156.1390294. http://portal.acm.org/citation.cfm?doid=1390156.1390294
128. Vinyals, O., Toshev, A., Bengio, S., Erhan, D.: Show and tell: a neural image caption generator. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3156–3164 (2015)
129. Weiss, K., Khoshgoftaar, T.M., Wang, D.: A survey of transfer learning. J. Big Data **3**(1), 9 (2016)
130. Werbos, P.J.: Backpropagation through time: what it does and how to do it. Proc. IEEE **78**(10), 1550–1560 (1990)
131. Wu, G., Kim, M., Wang, Q., Gao, Y., Liao, S., Shen, D.: Unsupervised deep feature learning for deformable registration of MR brain images. In: International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 649–656. Springer, New York (2013)
132. Wu, Y., Jiang, M., Lei, J., Xu, H.: Named entity recognition in Chinese clinical text using deep neural network. Stud. Health Technol. Inf. **216**, 624 (2015)
133. Wulsin, D., Blanco, J., Mani, R., Litt, B.: Semi-supervised anomaly detection for EEG waveforms using deep belief nets. In: 2010 Ninth International Conference on Machine Learning and Applications (ICMLA), pp. 436–441. IEEE, New York (2010)
134. Wulsin, D.F., Gupta, J.R., Mani, R., Blanco, J.A., Litt, B.: Modeling electroencephalography waveforms with semi-supervised deep belief nets: fast classification and anomaly measurement. J. Neural Eng. **8**(3), 36015 (2011)
135. Xing, Z., Pei, J., Keogh, E.: A brief survey on sequence classification. ACM SIGKDD Explorations Newsletter **12**(1), 40–48 (2010)
136. Xu, L., Ren, J., Yan, Q., Liao, R., Jia, J.: Deep edge-aware filters. In: International Conference on Machine Learning, pp. 1669–1678 (2015)
137. Yang, X., Kwitt, R., Niethammer, M.: Fast predictive image registration. In: Deep Learning and Data Labeling for Medical Applications, pp. 48–57. Springer, Cham (2016)
138. Yang, W., Chen, Y., Liu, Y., Zhong, L., Qin, G., Lu, Z., Feng, Q., Chen, W.: Cascade of multi-scale convolutional neural networks for bone suppression of chest radiographs in gradient domain. Med. Image Anal. **35**, 421–433 (2017)
139. Yang, X., Kwitt, R., Styner, M., Niethammer, M.: Quicksilver: fast predictive image registration–a deep learning approach. NeuroImage **158**, 378–396 (2017)
140. Zeng, H., Edwards, M.D., Liu, G., Gifford, D.K.: Convolutional neural network architectures for predicting DNA–protein binding. Bioinformatics **32**(12), i121–i127 (2016)
141. Zhang, S., Zhou, J., Hu, H., Gong, H., Chen, L., Cheng, C., Zeng, J.: A deep learning framework for modeling structural features of RNA-binding protein targets. Nucleic Acids Res. **44**(4), e32–e32 (2015)
142. Zhang, Q., Xiao, Y., Dai, W., Suo, J., Wang, C., Shi, J., Zheng, H.: Deep learning based classification of breast tumors with shear-wave elastography. Ultrasonics **72**, 150–157 (2016)
143. Zhao, Y., He, L.: Deep learning in the EEG diagnosis of Alzheimer's disease. In: Asian Conference on Computer Vision, pp. 340–353. Springer, Heidelberg (2014)
144. Zhou, J., Troyanskaya, O.G.: Predicting effects of noncoding variants with deep learning-based sequence model. Nat. Methods **12**(10), 931 (2015)

145. Zhu, J., Pande, A., Mohapatra, P., Han, J.J.: Using deep learning for energy expenditure estimation with wearable sensors. In: 2015 17th International Conference on E-health Networking, Application & Services (HealthCom), pp. 501–506. IEEE, New York (2015)
146. Zou, B., Lampos, V., Gorton, R., Cox, I.J.: On infectious intestinal disease surveillance using social media content. In: Proceedings of the 6th International Conference on Digital Health Conference, pp. 157–161. ACM, New York (2016)