Deep Learning in Smart Health: Methodologies, Applications, Challenges

Murat Simsek, Alex Adim Obinikpo, and Burak Kantarci

Abstract The advent of artificial intelligence methodologies pave the way towards smarter healthcare by exploiting new concepts such as deep learning. This chapter presents an overview of deep learning techniques that are applied to smart healthcare. Deep learning techniques are frequently applied to smart health to enable AI-based recent technological development to healthcare. Furthermore, the chapter also introduces challenges and opportunities in deep learning particularly in the healthcare domain.

Keywords Predictive analytics · Deep learning · Smart health · Medical imaging

1 Introduction

Urban and regional planning is an aspect of human endeavor that has expanded as man improves in knowledge and understanding. This expansion has seen tremendous improvement in the way and manner humans move about within and outside their immediate vicinity. This success was no doubt assisted by tools used for proper town planning of which maps and other location positioning services are part of. The need to constantly get the best within our community and outside our community has led to the improvement of the tools used for location positioning and other factors that contribute to a better-planned city [32]. This constant improvement

M. Simsek
School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, ON, Canada

Department of Astronautical Engineering, Istanbul Technical University, Istanbul, Turkey
e-mail: msimsek@uottawa.ca

A. A. Obinikpo · B. Kantarci (✉)
School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, ON, Canada
e-mail: aobin064@uottawa.ca; Burak.Kantarci@uottawa.ca

A. El Saddik et al. (eds.), *Connected Health in Smart Cities*,
https://doi.org/10.1007/978-3-030-27844-1_3

was helped and continuously being helped by improved technological output. Digitalization of our cities is seeing daily improvement with the aid of powerful devices embedded with sensors for data acquisition, environmental monitoring, digital transportation, health improvement, easy access to amenities and facilities, and overall service provision for everyone within the city. A digital city is more often than not called a smart city. A smart city has various components which are all interlinked with improved technology and the need to provide quality services to its citizenry [15]. One of this components is smart health.

Smart health is the use of high technological devices for improved and quality health delivery. In other words, it contains the use of smart devices, electronic health monitoring gadgets, web services all connected(or not) to a data hub where positive inferences could be made about an individual's health status or a community's health status. The ubiquitousness of smart health has made its development a welcome change; this coupled with the ever-growing production of technological tools has seen the demand for smart health applications go up in recent years. As an example, an individual could check his or her blood pressure with his or her mobile devices thanks to the embedded sensors and applications found on these devices [7]; it is also possible to check the weather or climate readings of an area within a city and to know which part of the city to avoid if the weather is not suitable for your health [41]; a medical professional can check his or her patients health condition using the application both of them share with the purpose of advising the patients should any emergency occurs and so on. These examples are just a few of the many advantages smart health has to offer and with the increase in technological advancements, better devices are being produced to cope with the demands of the smart health industry. These devices not only serve as health tools but they also serve as good data acquisition tools, in which case the generated data could be processed and useful inference can be made in the long run. This also makes Smart Health applications integral parts of smart cities development. However, for smart health to be a successful element within a smart city, it ought to be able to measure up to the level of growth when compared to other aspects of a smart city. In other words, smart health needs to advance with technology just like the other components of a smart city. For this to work, processing of a smart health dataset would require proper and improved techniques; this is somewhat becoming a research hurdle as the datasets are generated by different devices with various operating capabilities thus leading to datasets with varying output type and format. The question now becomes; how do we process these datasets effectively and efficiently considering the volume and format of these generated data in order to achieve the goals of smart health? To provide answers to this question and others like it, different methods were developed and proposed by various data scientists. These methods would further metamorphous into much broader techniques, the most popular of them being machine learning.

Machine learning (ML) is the ability of a machine to learn from inputs with the goal of producing powerful algorithms for decision making. With advancing technologies, comes different learning ways by which a machine learns. This learning evolution has led to the development of more sophisticated tools like

deep learning, extreme learning, etc. These tools have proven to be better to keep up with the aspect of new technological developments than conventional machine learning techniques. In fact, deep learning (one of the new tools) has a wide range of applications in smart health, specifically bioinformatics [36], medical imaging [18], disease prediction and analysis [40] to mention but a few. In the next section, we will talk about the improvement of the deep learning that is the next generation of machine learning.

2 Evolution of Deep Learning

The dataset generated by devices requires some form of processing for it to be useful. This processing was done using techniques that include the conventional machine learning algorithms. There two major types of machine learning algorithms; Supervised learning algorithm, where an input data with labeled responses are fed to the machine and the machine predicts the output (Support vector machines, decision trees, etc.); and unsupervised learning algorithms, which groups the input data into different classes or clusters based on certain characteristics (for example, K-means, DBScan, etc.). Machine learning algorithms use the features within a dataset to teach the machine how to identify patterns or specific characters like handwriting and speech [38]. The usefulness of machine learning algorithms in certain fields, for example, health care [51], computer vision [25], and so on [6], made them the "go to" tools for data processing and analysis. However, due to the increase in volume of datasets and the unstructured nature of data, these machine learning algorithms tend to face limitations in achieving the desired results. These and many other shortcomings lead to the development of a more computationally intensive and powerful learning technique called deep learning.

Deep learning algorithms have been described as a set of algorithms that think like the human brain [45]. A deep learning algorithm divides the dataset into layers and learn each layer, one by one, more like a "Divide and conquer" approach to problem solving. Deep learning techniques are gaining relevance as the year goes by due to the ease in which deep learning algorithms tackle problems in relative shorter time while consuming less memory. The development of deep learning was a gradual process borne out of the need to develop a machine that can deliver faster and work with high level of dimensions. This urge was given a boost when in 1958, Rosenblatt invented "perceptron." Hence, the first Artificial Neural Network (ANN) emerged and more development was begun [43]. The goal is to model the machine to think like the human brain and learn on its own. ANNs were used to do tasks that ordinarily could have been difficult for the computer without certain defined rules. However, as the year went by, improvements were needed to help the early invention keep up with the changing trend in computation. One of these improvements was introduced by Ivakhnenko [24] where he developed an algorithm for supervised deep feed-forward network; in this algorithm, layers grow incrementally, then trained using regression analysis and trimmed using validation sets to give effective

output. Then in 1970, Linnainmaa developed the back propagation technique which is considered as the backbone of deep learning. Fukushima [17] while building his deep neocognition architecture introduced and added weights to convolutional neural networks. This created a gradient-based deep learning algorithms. All these were done in order to find a better way to train multiple layered network. To further expand on previous stated techniques, LeCun et al. [29], combined the back Propagation (BP) algorithm with a deep neural network in his research on hand written zip code recognition which proved successful. This further led to other useful ways to properly train a multilayered perceptron and further develop deep learning algorithm as seen in [3, 11, 19, 20, 46]. These historical developments of deep learning can be summed up into two major characteristics of deep learning. The first characteristic is the ability to discover hidden structures within large datasets using the back propagation algorithm which tells a machine how it should handle its parameters used in the computation of a layer and its successor. This argument sometimes lead to deep learning been termed as an example of representation learning. Another characteristic is deep learning adjust to unforeseen circumstances even if it has no knowledge of the rule governing such problems before hand. This is a necessary characteristic since the machine cannot be trained with loose data. Loose data occurs when proper problem description cannot be delivered to the machine thus leading to inadequate data that could have helped the machine make meaningful inference.

Deep learning is an effective tool in all fields through these two properties, especially healthcare. The diverse applications of deep learning in healthcare have evolved over the years and would be discussed in details in subsequent sections of this chapter. However, deep learning methods in healthcare mainly discussed in the following section.

3 Deep Learning-Based Methods

The following subsections will mention about the various type of deep learning methods that are mostly applied on smart health technologies. Fundamental notations which are required to understand mathematical relations in the remaining part of the section can be seen in Table 1, which has been adopted from [39].

3.1 Deep Feed-Forward Networks

Deep feed-forward primarily aims to approximate a function $f*$ by defining a mapping $y = f(x, \theta)$ which learns the value of θ in order to get a best approximation. To get the final value of y, several iterations are done within the layers. A typical feed-forward neural network consists of three fundamental layers. The input layer and the output layer should equal to the dimension of the input

Table 1 Basic notations used in the chapter

Notations	Definition
x	Samples
y	Outputs
v	Visible vector
h	Hidden vector
q	State vector
W	Matrix of weight vectors
M	Total number of units for the hidden layer
w_{ij}	Weights vector between hidden unit h_j and visible unit v_i
t_i	Signals
a_{ik}	Mixing weights
S_j	Binary state of a vector
s_i^q	Binary state assigned to unit i by state vector q
Z	Partition factor
b_j	Biased weights for the j-th hidden units
a_i	Biased weights for the i-th visible units
z_i	Total i-th inputs
v_i	Visible unit i
w_{kj}^2	Weight vector from the k-th unit in the hidden Layer 2 to the j-th output unit
w_{ji}^1	Weight vector from the j-th unit in the hidden Layer 1 to the i-th output unit
W_{ji}^1	Matrix of weights from the j-th unit in the hidden Layer 1 to the i-th output unit
$E(q)$	Energy of a state vector v
σ	Activation function
$P_r(q)$	Probability of a state vector q
$E(v, h)$	Energy function with respect to visible and hidden units
$pdf(v, h)$	Probability distribution with respect to visible and hidden units
$(A(n(t\|m)))$	Entropy of the posterior

space and output space of the model. The hidden layer can be single or multiple according to the complexity of the model. Training process is required to ensure that $f(x)$ matches $f*$. In this case every sample in x has an accompanying attribute in $y \approx f*(x)$. In order to get the better approximations of $f*(x)$, the algorithm develops and uses the hidden layers. The hidden layers are the iterative computations which are done before the final result is sent to the output layer.

Mathematically, a basic feed-forward network with single hidden layer can be described in Eq. (1). Let $y_1, \ldots, y_k, ..y_M$ be M outputs for N dimensional input x and H_1 the number of neurons in the single hidden layer, then general output y_k could be given as follows:

$$y_k(x, w) = \sigma \left(\sum_{k=1}^{M} w_{kj}^{(2)} \sum_{j=1}^{H_1} \sigma_j \left(\sum_{i=1}^{N} w_{ji}^{(1)} x_i \right) + w_{j0}^{(1)} \right) + w_{k0}^{(2)} \tag{1}$$

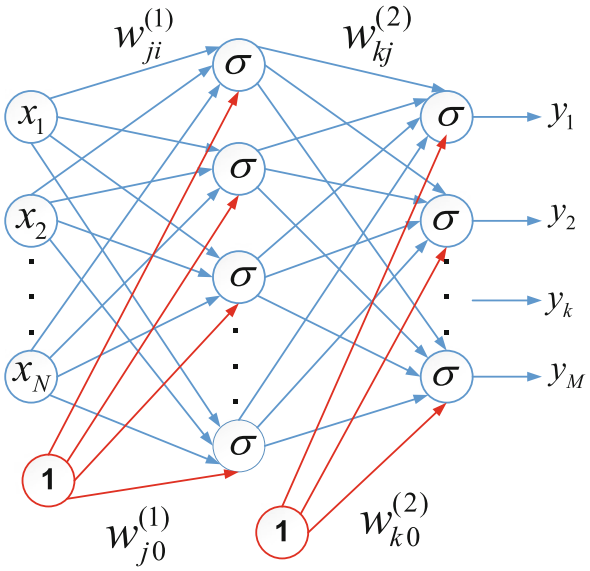

Fig. 1 Basic feed-forward network with single hidden layer

where $w_{kj}^{(2)}$ and $w_{k0}^{(2)}$ are weights associated with output layer; $w_{ji}^{(1)}$ and $w_{j0}^{(1)}$ are weights associated with hidden layer. Nonlinear modeling can be possible through nonlinear activation function σ.

Deep feed-forward network in Fig. 2 has more hidden layer than basic feed-forward network in Fig. 1. Number of hidden layer H can be shown in Fig. 2. The more hidden layer provides more processing capability for Deep Networks.

3.2 Linear Factor Models

Given a latent variable h and a real variable x, and if

$$h \approx p(h) \tag{2}$$

Then we can define a linear model as Eq. (3).

$$x = wp(h) + b + noise \tag{3}$$

where $p(h)$ is a factorial distribution, b is the bias, and w is the weight, and the noise is independent over all dimensions and Gaussian dependent.

Equation (3) is the base linear factor model where other forms will be derived from as we will see in later subsections.

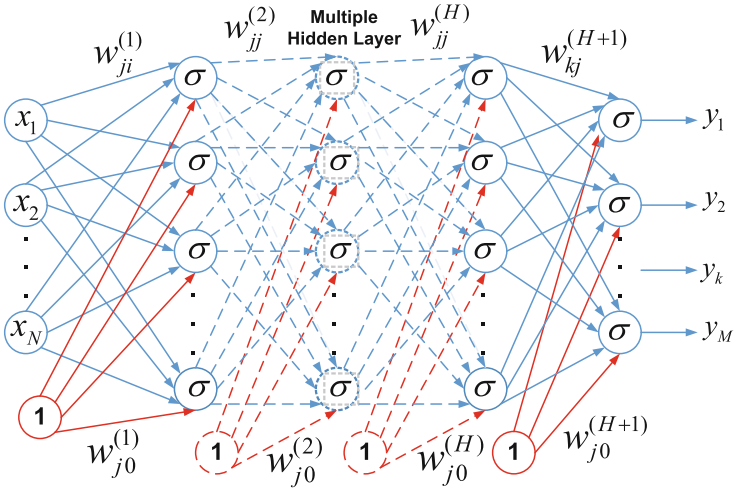

Fig. 2 Deep feed-forward network with H number of hidden layers

3.2.1 Probabilistic Principal Component Analysis (PCA)

The first variant of the linear factor model is the PCA. In order to utilize Eq. (3), the PCA allows the noise variation to occur when approximating the latent variable h before the real valued variable x is observed. That is,

$$h \approx N(h; 0, I) \tag{4}$$

With variables x_i assumed to be conditionally independent with respect to h, we get the following:

$$x \approx N(x; b, WW^T + \psi) \tag{5}$$

In this case, x is a multivariate normal random variable, ψ is the covariant matrix given as $\psi = diag(\sigma^2)$. We can define σ^2 as per-variable variance and it is represented in vector form $[\sigma_1^2 + \sigma_2^2 + \ldots + \sigma_n^2]^T$. Substituting this into Eq. (4) and adjusting the initial model equation (5), we obtain

$$x \approx N(x; b, WW^T + \sigma^2 I) \tag{6}$$

Equation (6) is the model for the PCA and $WW^T + \sigma^2 I$ is the covariance of the variable x. Decomposing Eq. (6) further gives

$$x = Wh + b + \sigma z \tag{7}$$

where $z \approx N(z; 0, \sigma I)$ is the introduced Gaussian noise.

3.2.2 Independent Component Analysis (ICA)

The second variant of linear factor models is the ICA. The ICA divides observed signals into many independent non-Gaussian parts, then fuse them to become the observed/input data.

Suppose, we have T signal divided into $T = (t_1, \ldots, t_n)^T$ and a random vector x given as $x = (x_1, \ldots, x_m)^T$, then the input data can be of the form

$$x_i = a_{i,1}t_1 + \ldots + a_{i,k}t_k + \ldots + a_{i,n}t_n \tag{8}$$

where $a_{i,k}$ is the mixing weights.

Now, if we let $x_1, x_2, \ldots, x_m$ be the set of binary variables from m monitors with a corresponding $y_1, y_2, \ldots, y_n$ of n sources, then we have

$$x_i = \vee_{j=1}^{n}(g_{ij} \wedge y_i), i = 1, 2, \ldots, m \tag{9}$$

where $\vee$ is Boolean "OR" and $\wedge$ is Boolean "AND." Equation (9) is called the binary ICA model and the monitors and sources are in binary form.

3.3 Autoencoder

An autoencoder [22, 49] is a fully connected neural network which consists of three layers such as input, hidden, and output. The autoencoder can be decoupled into two separate parts: an encoder $h = f(x)$ and a decoder $r = g(h)$, both sharing the layer which is often referred to as base vector as depicted in Fig. 3. If the autoencoder successfully learns to place $g(f(x)) = x$ everywhere, then it becomes irrelevant

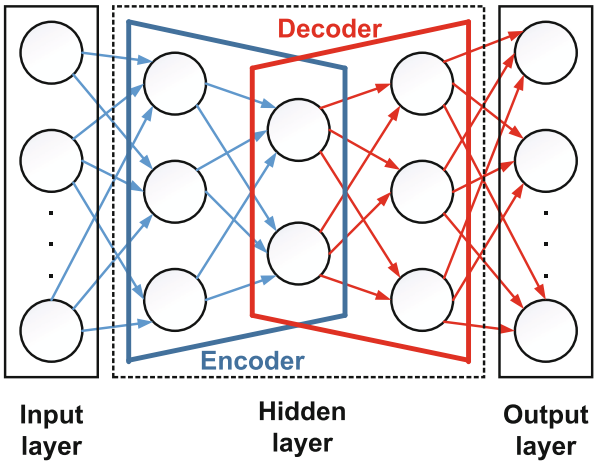

Fig. 3 Autoencoder network

because autoencoders are usually made just to be able to copy the original a bit but not perfectly. They are viable tools for dimensionality reduction, feature learning, and generative modeling.

There are various types of autoencoders which will be described briefly:

- Undercomplete Autoencoders: have a smaller dimension for hidden layer compared to the input layer. This helps to obtain important features from the data. Objective function in Eq. (10) minimizes the loss function by penalizing the $g(f(x))$ for being different from the input x. Objective is to minimize the loss function by penalizing the $g(f(x))$ for being different from the input x.

$$L = |x - g\,(f\,(x))| \tag{10}$$

  where L is the loss function.
- Regularized Autoencoders: This kind of autoencoder trains any type of architecture by choosing the code dimension and other properties based on the complex nature of the dataset to be modeled [4]. The regularized autoencoder uses a loss function most times to give the model a leeway in exploiting other encoder properties rather than limiting itself to just the ability to copy inputs to outputs.
- Sparse Autoencoders: have hidden neurons greater than input neurons. Sparsity constraint is introduced on the hidden layer which is to prevent output layer exactly copy to input data. They can still discover important features from the data. The sparse autoencoder's training requirement involves the imposition of a sparsity penalty $\Omega(h)$ on the hidden layer h, together with the reconstruction error equation (11).

$$L = |x - g\,(f\,(x))| + \Omega\,(h) \tag{11}$$

  Again, sparse autoencoders are sometimes used to learn features during a classification task.
- Denoising Autoencoder: is a stochastic autoencoder as a stochastic corruption process to set some of the inputs to zero [49]. Denoising refers to intentionally adding noise to the input before providing it to the network. Denoising autoencoders minimizes the loss function equation (12) between the output node and the corrupted input $\tilde{x}$ which is obtained by adding noise to the input.

$$L = |\tilde{x} - g\,(f\,(x))| \tag{12}$$

- Contractive Autoencoders: is another regularization technique like sparse autoencoders and denoising autoencoders. It can be considered to have a robust learned representation which is less sensitive to small variation in the data. Robustness of the data representation is ensured by applying Frobenius norm of the Jacobian matrix as a penalty to the loss function [42]. This penalty for the hidden layer is calculated with respect to input. Once penalty term in Eq. (11) is changed by Eq. (13), then Eq. (14) is used as Frobenius norm in Eq. (13). Hence, loss

function can be obtained by changing the Eq. (11).

$$\Omega(h) = \lambda \left\| J_f(x) \right\|_F^2 \tag{13}$$

$$\left\| J_f(x) \right\|_F^2 = \sum\nolimits_{ij} \left(\frac{a h_j(x)}{\partial x_i} \right)^2 \tag{14}$$

In this case, the penalty $\Omega(h)$ is called the Frobenius norm.

The contractive autoencoders are trained to discourage any form of perturbation of their input values, so they try to map the neighborhood of input values into a much smaller neighborhood of output values.

3.4 Convolutional Neural Network (CNN)

Convolutional Neural Net is a more powerful deep learning technique to improve the performance for current visual recognition tasks [1, 12]. CNN's structure can be determined in terms of the size, quality, and type of dataset. They are neural networks that use mathematical convolutions besides general matrix multiplications. CNNs consist of multiple receptive layers that process portions of the input image. The outputs of CNNs are arranged in such a way that it creates some form of overlapping in the input area, in order to obtain a higher-resolution representation of the original image. The same procedure is run for every layer that is present in the network. Moreover, the goal of CNNs is to learn data-specific kernels instead of predefined kernels.

3.4.1 CNN Architecture

CNNs will utilize a series of convolutions and pooling operations during which the features are detected. The fully connected layers will work as a classifier using these extracted features. All operations in CNN are summarizes in Fig. 4. It is worthy to note that though, any CNN might have a few amount of convolutional layers coupled with pooling layers, it is optional for it to have fully connected layers.

- Convolutional Layer: Convolution is one of the main building blocks of CNNs. The convolution is used for the mathematical combination of two functions and the result is a function as well. The convolution is executed by sliding the filter over the input. A matrix multiplication is performed for every location and sums the result onto the feature map. The convolutional layer has a $m \times m \times r$ input image, where m is the height and width of the image and r is the number of channels. It has k filters (or kernels) and size of each is $n \times n \times q$. n in this case, is smaller than the image dimension. q could either be the same as the amount

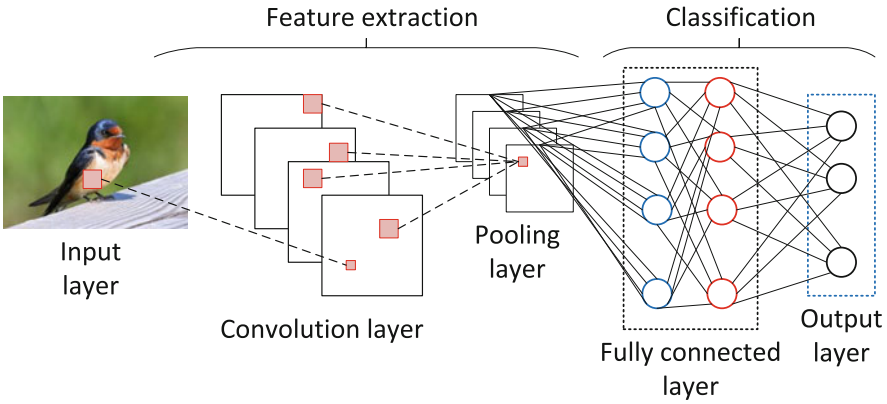

Fig. 4 Architecture of a CNN

of channels r or it might be smaller for every kernel. The filters' size leads to connected structures where each structure is linked with the image to constitute k feature maps of size $m - n + 1$.

- Pooling Layer: After a convolution layer, a pooling layer is commonly added between CNN layers. The main function of pooling layer is to reduce the dimensionality to satisfy lower the number of parameters and computation in the network. Hence, the training time is reduced and it can be possible to control overfitting. Pooling layer sub-samples their input by applying a maximum operation to the result produced by each filter or kernel. One major property of pooling is to generate a fixed size output matrix that is required for classification. The most favorite pooling is max pooling, which takes the maximum value in each window.
- Fully Connected Layer: After the convolution and pooling layers, the last part is required to be fully connected regular Neural Network to classify input images. Neurons in a fully connected layer are connected to the activation functions in the previous layer.

3.5 Deep Belief Network (DBN)

The DBN consists of stochastic binary unit layers where each connected layers have some weight. The DBN has multiple layers of latent variables that is connected between layers [21, 23]. Though with these connections amongst layers, there exists no visible connections amongst units that are within a particular layer. Again, the DBN learns to do a new construction of its inputs, and thereafter train them for classification tasks. One important feature of the DBN is that, they are learned

one particular layer per time using the greedy scheme. The DBN has the following properties:

- When learning the generative weights, a layer by layer approach is used which determines how each variable in a layer depends on variables in another layer that is above it.
- After learning each latent variable, their values are inferred using a single pass which begins with an observed data in the least layer.

Furthermore, suppose, we have the visible units v, the hidden units, h that are conditionally independent, the weights W, that is learned by a restricted Boltzmann machine, then the probability of generating a visible vector v, is

$$p(v) = \sum_{h} p(h\backslash W)p(v\backslash h, W) \tag{15}$$

where $p(h\backslash W)$ is the prior distribution over hidden vectors and $p(v\backslash h, W)$ is the posterior distribution over visible vectors.

In one sense, if a DBN has just one hidden layer, it is called a restricted Boltzmann machine (RBM). In this case using a constructive divergence method, we can train the first RBM which subsequently leads to the training of DBN after certain number of iterations.

3.6 Boltzmann Machine (BM)

BM is a symmetrically connected network of neuron-like units (Fig. 5) that make binary stochastic decisions [2]. The learning algorithms of BMs allow them to fully discover useful and important features that portrays complex regularities in datasets. BMs are mostly used to solve search and learning problems.

To get a better understanding, assume a unit i has an opportunity to always update its binary state at any given time; it computes its total input as seen in Eq. (16), below.

$$z_i = b_i + \sum_{j} s_j w_{ij} \tag{16}$$

where w_{ij} refers to the connection weight between units i, j, and s_j is 1 if j is on or 0 when j is off. The probability that unit i comes on is given as

$$P_r(s_i = 1) = \frac{1}{1 + e^{-z_i}} \tag{17}$$

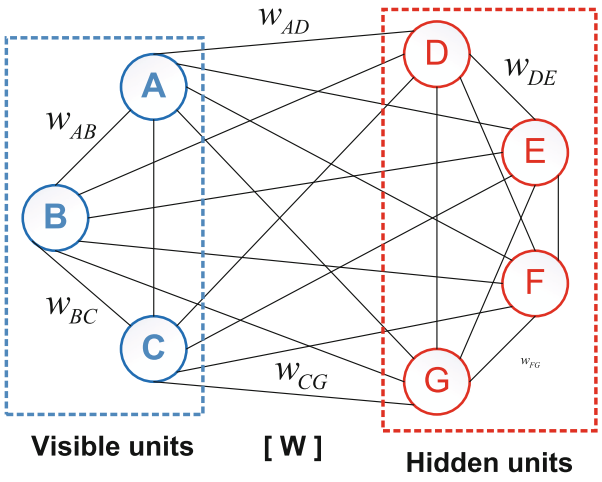

Fig. 5 Boltzmann network

Now, if each unit is updated sequentially, the network in the probability of a state vector v can be obtained in Boltzmann distribution as follows:

$$P_r(q) = \frac{e^{-E(q)}}{\sum_u e^{-E(u)}} \tag{18}$$

where $E(q)$ is the energy of state vector q and defined as

$$E(q) = -\sum_i s_i^q b_i - \sum_{i<j} s_i^q s_j^q w_{ij} \tag{19}$$

where s_i^q refers to the binary state assigned to unit i by state vector q.

Should any of these connections weights be selected in a way that the energies of each state vectors represent their costs, then we can view the stochastic nature of a BM as a means of exiting from an inappropriate local optima while it continues its search for low-cost solutions.

When learning a BM, it can be done with the hidden units or without them. There are special cases or types of BM, out of which two are highlighted below.

Mean Field Boltzmann Machines. This kind of BM uses mean field units which possesses deterministic values between 0 and 1, and they are used to compute the main value for a unit's state based on the current states of the other units.

*High-order Boltzmann machines.*In this type, the structure and the rule for learning encourage the use of energy functions that are complicated.

3.7 Restricted Boltzmann Machines (RBM)

In an RBM, there exists layers of visible and hidden units with intraconnections within these layers (that is, no hidden-hidden nor visible-visible connections) [22, 28].

With the hidden units (h) being independent conditionally on the visible (v) vector, the unbiased samples from $\langle s_i s_j \rangle_{data}$ can be obtained in one single step. In order to take samples from $\langle s_i s_j \rangle_{model}$ requires a number of iterations with alternating activities between updating the hidden units and the visible units in parallel times [44].

Mathematically, the energy function of an RBM with hidden and visible units consisting of $W = (w_{ij})$ (where W is the matrix of weights) associated with the connection between hidden unit h_j and visible unit v_i, can be written as

$$E(v, h) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_i \sum_j v_i w_{i,j} h_j \tag{20}$$

with a probability distribution of

$$pdf(v, h) = \frac{1}{Z} e^{-E(v,h)} \tag{21}$$

After learning is finished for one hidden layer, the activity vectors of the hidden units can be treated as "data" to train another RBM. This particular computation can be repeated as many times as possible in order to learn as many hidden layers as needed. After learning many hidden layers, the entire network can be seen as a single but multi-layered generative model where additional hidden layers contribute to the improvement of the lower bound (Fig. 6).

Learning hidden layers one at a time has been seen as a very efficient way to learn and understand deep neural networks that possess multiple hidden layers with quite a number of weights. The learning might be unsupervised but the highest features are generally useful for classification purposes.

3.8 Variational Autoencoders (VAE)

These autoencoders use learned approximations to make inference. They are trained mainly using gradient-based methods. To obtain a sample from an already built model, the VAE chooses a sample t from the distribution $p_{model}(t)$ and runs it through a generator network $g(t)$. After which, the random variable m is chosen randomly from the distribution $p_{model}(m; g(t)) = p_{model}(m|t)$. While training is

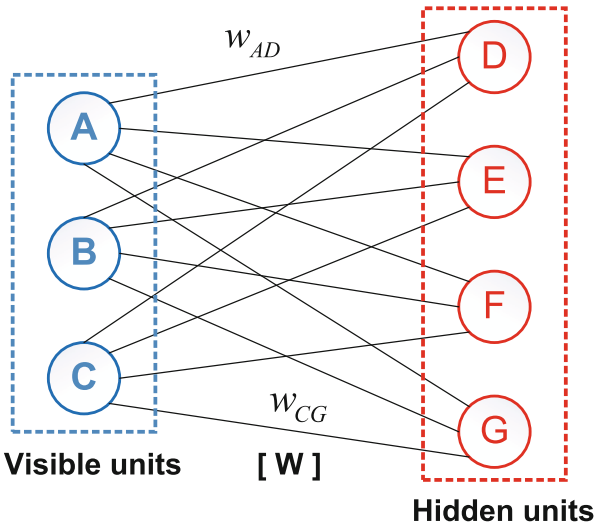

Fig. 6 Restricted Boltzmann network

going on, the generated approximate inference network $n(l)$ is used to derive t while $p_{model}(m|t)$ becomes the encoder of the network. In other words a VAE could be trained properly if the variational lower bound $D(n)$ that is associated with the random variable m is maximized such that

$$D(n) = E_{t \approx n(t|m)} log p_{model}(t, m) + A(n(t|m)) \tag{22}$$

where $A(n(t|m))$ refers to the entropy of the posterior.

If n is a Gaussian distribution with an additive noise added to its predicted mean value, then a maximum entropy value would increase the value of the standard deviation of the noise. Another way of saying this is, the entropy value allows the variational posterior to place a steep probability mass function on a number of t items which would have produced m rather than reducing it to just a point estimate of the most probable value.

While some approaches to VAE infer the value of n through an optimization algorithm, the main goal however is to train any parametric encoder to produce parameters of n. Thus, if t is a continuous variable, then carrying out a back propagation on the samples of m will give a gradient with respect to the encoder(θ).

One important feature with the VAE is that it is very possible to train a combined parametric encoder and generator network function, thus giving the model the ability to learn a predictable coordinate system which the encoder captures.

3.9 Auto-Regressive (AR) Model

An auto-regressive (AR) model is utilized to predict future characteristics based on past values. AR is also used for forecasting when there is some correlation between values in a time series and the values [27, 35]. Since AR requires past data to model the behavior, the name auto-regressive is related to "self." The process is very similar to a linear regression of the data in the current series opposing one or more past values in the series.

The AR process is a stochastic process, which has degrees of uncertainty or randomness built in. The randomness means that AR might be able to predict future trends accurately in terms of past data, but this accuracy is never going to get %100. Generally, the process can be close enough to the desired response. AR models are also called conditional models or Markov models.

An $AR(p)$ model is an auto-regressive model where specific lagged values of y_t are used as predictor variables. The value for "p" is called the order. $AR(1)$ indicates the first-order auto-regressive process. The response in a first-order AR process at some point in time t is related only to time delayed response. The high order AR process is related to the corresponding time delayed response data. $AR(p)$ model formulation is given as follows:

$$y_t = \delta + \varphi_1 y_{t-1} + \varphi_2 y_{t-2} + \ldots + \varphi_p y_{t-p} + A_t \tag{23}$$

where A_t is white noise and φ indicates constant values. Moreover y_{t-1} indicates the first-order time delayed response.

δ in (24) is seen in (23).

$$\delta = \left(1 - \sum_{i=1}^{p} \phi_i\right) \mu \tag{24}$$

where μ is the process mean.

3.10 Nonlinear Auto-Regressive (NAR) Neural Networks

Nonlinear auto-regressive (NAR) neural networks are mostly suitable for prediction and forecasting [14, 16]. The output of NAR neural network is generated by regarding different ordered delayed outputs. Hence previous output data are used for prediction of the future output. Nonlinear activation functions provide a nonlinear relationship between time delayed data and the current time output. The structure of NAR neural network can be seen in Fig. 7.

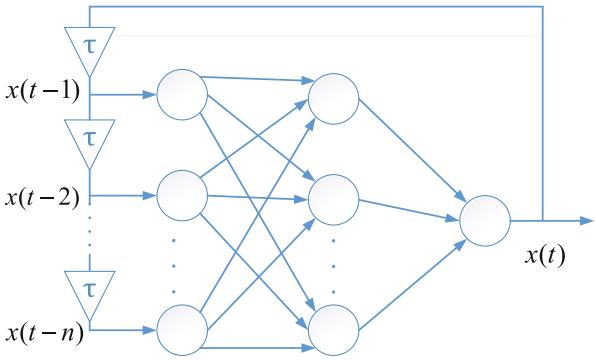

Fig. 7 Auto-regressive Neural Network (ARNN)

4 Applications

Health care systems are being revolutionized due to rapid growth in technology. The applications of various technological innovations have contributed immensely to the growth of quality healthcare delivery. As stated in Sect. 1, the growth of smart health can be attributed to the ever-growing ecosystem in the technological space. And as this growth increases, comes the burden of specialization in terms of proper algorithms for problem solving; ability to synchronize the data acquired with the intended technique for data analysis; how and when to use a particular technique; what techniques should be used for effective results, amongst many others. These are some of the many problems plaguing the smart health industry. However intimidating these problems might seem, research has been conducted to provide up-to-date solutions to these problems. Most research was conducted according to various themes in the industry which includes but not limited to data acquisition, data processing, data analytics, and learning techniques. That said, for any meaningful progress to be made, there is always a base case. The base case for smart health care is a 2-prong one: data acquisition and data analytics. Data acquisition can occur in various forms with different output format. This often leads to data heterogeneity when various devices are used to acquire data for experimental purposes. The format of the acquired data determines what kind of techniques that would be used for analytics. Deep learning, one of the tools used for data analytics has proven to be an effective tool due to its accuracy, runtime, and usage with almost any kind of data. Deep learning can make classify, cluster, and predict possible by getting signals, or structures in datasets. When deep network is trained, it can be used for prediction about the data. The prediction error of it can be measured regarding with the training set. What's more is that deep learning has a range of applications in the smart health industry. In this section of the chapter, we shall be looking at a few applications of deep learning techniques in smart health. These applications shall be discussed based on three health categories; bioinformatics, medical imaging, and predictive analytics.

4.1 BioInformatics

Bioinformatics is one area in the health industry that has been receiving lots of attention in recent years. This is not far-fetched since the health industry is currently experiencing massive digitization and the utilization of this massive technological growth is evident. Deep learning techniques are being used widely in the bioinformatics area of health care. Specific cases are summarized as follows. Understanding protein structures is a key research topic in the medical area. This is due to the fact that protein structures are key components in the understanding of the functions of proteins. The authors in [36] proposed a deep learning technique called sparse autoencoder for sequence-based determination of the distance between two neighboring $c\alpha$ atoms represented by the angles between $C\alpha_{i-1} - C\alpha_i - C\alpha_{i+1}$ (θ) and $C\alpha_i - C\alpha i + 1$ (τ). They believed that accurate predictions of these angles can appropriately give a more accurate distance. Also, the predicted α and τ values could be used to construct local structures with good accuracies. In the same vein, the authors in [5] show that DNA- and RNA- binding proteins sequence specificities can be derived from experimental data. They called this approach "DeepMind." This method uses the convolutional neural network for training the acquired data and then used back propagation to compute the derivative of all parameters that are in the model. The result obtained from Deepmind was better than other methods used in this particular application area. Also, Lee and Yoon [30] used a deep learning-based technique-restricted Boltzmann machines to predict protein secondary structure. The demand for protein secondary structure predictions is on the increase in the protein discovery sphere of bioinformatics. The authors generated multiple layers for the intended network from a dataset of 1230 protein chains, used the RBM for training, analysis, and prediction. The outcome of the prediction was evaluated using the SOV scoring functions. Their proposed method generated a result of 80.7% accuracy when tested with an independent test dataset. In their work, Leung et al. [31] used deep neural network to create a model that can predict splicing patterns in individual tissues as well as individual differences found in splicing patterns across tissues. Zhang et al. [52] with a view to model the structural binding for RNA-binding proteins used a deep learning framework. The developed learning framework constructed a representation for the specificities of the RNA-binding proteins and also predict new binding sites within the RNA being studied. Motivated by the need to accurately recognize gene expressions, the authors in [9] used D-GEXT (feed-forward neural network-based technique) to determine the expressions of certain target genes. The result from this experiment showed that D-GEX outperforms other methods by 6.57%. Aside gene classifications and expressions, deep learning is also being used in cancer research. Specifically, to enhance diagnosis of cancer, Fakoor et al. [13] proposed a method that is based on two deep learning techniques (principal component analysis and autoencoder). Their proposed method was able to detect and classify cancer types accurately from different cancer datasets. In similar vein, Danaee et al. [10] used the stacked denoising autoencoder for feature extraction and classification of cancer types from

high dimensional gene expression dataset with a view to accurately detect cancer. Again, a deep learning-based model was proposed by Wei et al. [50] for breast cancer image classification. The deep learning technique used for this research was convolutional neural network where class and sub-class labels of breast cancer are labeled in such a way that the distance between features in breast cancer images are restricted within a certain threshold. This method produced a classification accuracy of 97%. Similarly, Liu et al. [34] developed "XmasNet" a convolutional neural networks-based classification technique for prostate cancer identification.

4.2 Medical Imaging

While deep learning has a presence in bioinformatics, its presence is also found in medical imaging. Over the years, scientist and health care professionals have been seeking for new ways to process and analyze medical images properly and in a timely manner. And since deep learning surfaced, various deep learning-based techniques have been developed. These techniques have produced results that have outperformed the traditional image processing techniques. A case specific example is the use of deep learning algorithm for the detection of melanoma (a deadly form of skin cancer). In order to detect this tumor, the algorithm learns more about the features of the disease from a dataset containing medical images and makes appropriate predictions. Similarly, Li et al. [33] developed a method based on deep learning (CNN) for the detection of mitotic cells from pathological slides. This is done by first creating a deep segmentation of the mitosis region and there after designing a deep detection network for the localization of the mitosis region. The results from this showed a better F-score when compared to other methods. Aside tumor detection, deep learning is also being used to track tumor growth and development by generating probability heat maps which provides various information on the shape, size, density, and location of tumors [47]. In [48], the authors used deep learning-based technique called the stacked autoencoder to identify and categorize organs in MRI images from unlabeled and unstructured dataset. This they did with the hope of developing a working technique for organ identification within abnormal datasets. This technique achieved an accuracy of 96%. In order to be able to detect diabetic retinopathy early, Gulshan et al. [18] developed a deep learning algorithm that detects diabetic retinopathy early using images in the retinal fundus photographs. The deep learning algorithm used was the CNN and it had an accuracy of 97% in detecting this deadly case of diabetes. For brain lesion segmentation, Kamnitsas et al. [26] proposed a CNN-based technique to overcome earlier computational burden experienced during brain lesion segmentation; this technique is 11 layers deep and has proven to be an effective scheme when run on the MRI dataset of patients with brain injuries and brain tumors. Using non-medical image database, the CNN was used to identify various types of pathologies present in chest X-ray images; the idea was to use the algorithm/result to prove that deep learning could be applied to databases that are

all non-medical and still produce useful results and this experiment yielded a 93% accuracy [8].

4.3 *Predictive Analytics*

Predictive analytics is the use of past data coupled with some computations or analytical tools to predict an event. In the medical case, it is the use of a medical history or health care history to predict the outcome of an event. Predictive analytics has been an ongoing activity in recent years as successful prediction can help avoid adverse health condition of a patient or help reduce the effect of an outbreak. Various techniques have been employed in predictive analytics, of which deep learning is one of them. Some of the applications of deep learning in this aspect of health care are highlighted in the following sentences. Miotto et al. [37] proposed a method called Deep Patient, which used denoising autoencoder for deep feature learning and EHR data extraction with the view to properly facilitate clinical prediction of patients' health status. In similar vein, Pham et al. [40] developed DeepCare whose sole purpose is to read medical records and make predictions of future health outcomes. DeepCare uses long short-term memory for this purpose. Both Deep Patient and DeepCare showed a high performance rate with regard to disease predictions.

Table 2 shows a summary of the major applications of deep learning and their associated techniques.

Table 2 A summary of deep learning applications in smart health

Smart health theme		Deep learning technique used
Bioinformatics	Protein Structure prediction	Sparse autoencoder [36] CNN [5] RBM [30]
	Gene expression	DNN [31, 52] Feed-forward network [9]
	Cancer detection and identification	PCA,Autoencoder [13] Denoising autoencoder [10] CNN [34, 50]
Medical imaging	Tumor detection	CNN [33, 47]
	Brain lesion	CNN [26]
	Organ Identification	Stacked Autoencoder [18, 48]
Predictive analytics	Patients health prediction	Denoising autoencoder [37, 40] Auto-regressive NN [16]

5 Challenges in Deep Learning

Deep learning algorithms require huge amount of data to enable them perform excellently. The more perfection you want; the more data you will need to feed to the machine to enable the algorithms produce models that are powerful enough for your needs. This is mostly the bane of deep learning. And with big data comes the huge amounts of parameters required to get the deep learning algorithm properly tuned. In most cases, this huge amount of data is not available and whenever they are, it is mostly not enough as such, researchers are expected to augment the learning process through approximation.

Another challenge is the issue of overfitting. This is usually common in neural networks where there is a huge difference in the error that occurs when training a dataset and that which occur when a new dataset is introduced. This is an issue since the reason a model is being trained is to be able to perform well when it is used on a new dataset rather than the one it was developed with.

Again, deep learning usually requires huge resource deployments for it to perform excellently well. That is, the more powerful your computing resources are, the more likely you are to get a more effective result from the deep learning algorithm. Aside computing resources, you would also require a huge amount of storage capabilities to train models effectively. Also, deep learning algorithm requires more time to train a dataset than the usual machine learning techniques.

Furthermore, deep learning algorithms are problem specific. That is, when a model is trained for a particular problem, it is usually difficult to tweak it for another kind of problem. This lack of flexibility is an issue because, it would lead to a waste of time to retrain and redevelop a new model for a seemingly similar problem.

6 Conclusion

Deep learning is proving to be an emerging and usable technique in smart health processing and applications. Even with its challenges, its use has been widely accepted in smart health.In the beginning of this chapter, we discussed briefly the emergence of smart city and its links to smart health. We also talked about the link between smart health and machine learning in the introductory section of this chapter. The recent transition from machine learning to deep learning was also discussed in Sect. 2, where we briefly highlighted the deep learning development timeline and evolution. Then, we introduced the basic deep learning techniques (feed-forward networks, autoencoders, linear factor models, convolutional neural networks) that are been used in smart health along with their major formulations. Furthermore, we highlighted the applications of deep learning techniques in smart health from cancer diagnosis to health status predictions. These applications were divided along the lines of bioinformatics, medical imaging, and predictive analytics. Lastly, the challenges of deep learning were discussed in the last section of this chapter.

References

1. A.H. Abdulnabi, G. Wang, J. Lu, K. Jia, Multi-task CNN model for attribute prediction. IEEE Trans. Multimedia **17**(11), 1949–1959 (2015)
2. D.H. Ackley, G.E. Hinton, T.J. Sejnowski, A learning algorithm for Boltzmann machines. Cogn. Sci. **9**(1), 147–169 (1985)
3. I.N. Aizenberg, N.N. Aizenberg, G.A. Krivosheev, Multi-valued and universal binary neurons: learning algorithms, application to image processing and recognition. Lecture Notes Comput. Sci. **1715**(4), 306–316 (1999)
4. G. Alain, Y. Bengio, S. Rifai, Regularized auto-encoders estimate local statistics, in *Proc. CoRR* (2012), pp. 1–17
5. B. Alipanahi, A. Delong, M.T. Weirauch, B.J. Frey, Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. Nat. Biotechnol. **33**(8), 831 (2015)
6. E. Alpaydin, *Introduction to Machine Learning* (MIT Press, Cambridge, 2014)
7. M.M. Baig, H. Gholamhosseini, Smart health monitoring systems: an overview of design and modeling. J. Med. Syst. **37**(2), 9898 (2013)
8. Y. Bar, I. Diamant, L. Wolf, H. Greenspan, Deep learning with non-medical training used for chest pathology identification, in *SPIE Medical Imaging*, vol. 9414 (2015), pp. 94140V-1–94140V-7
9. Y. Chen, Y. Li, R. Narayan, A. Subramanian, X. Xie, Gene expression inference with deep learning. Bioinformatics **32**(12), 1832–1839 (2016)
10. P. Danaee, R. Ghaeini, D.A. Hendrix, A deep learning approach for cancer detection and relevant gene identification, in *Pacific Symposium on Biocomputing* (World Scientific, Singapore, 2017), pp. 219–229
11. A. De Carvalho, M.C. Fairhurst, D.L. Bisset, An integrated Boolean neural network for pattern classification. Pattern Recogn. Lett. **15**(8), 807–813 (1994)
12. L. Deng, O. Abdelhamid, D. Yu, A deep convolutional neural network using heterogeneous pooling for trading acoustic invariance with phonetic confusion, in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (2013), pp. 6669–6673
13. R. Fakoor, F. Ladhak, A. Nazi, M. Huber, Using deep learning to enhance cancer diagnosis and classification, in *Proceedings of the International Conference on Machine Learning*, vol. 28 (2013)
14. R. Fan, F.-L. Zhang, M. Zhang, R.R. Martin, Robust tracking-by-detection using a selection and completion mechanism. Comput. Visual Media **3**(3), 285–294 (2017)
15. M. Finger, M. Razaghi, Conceptualizing "smart cities". Informatik-Spektrum **40**(1), 6–13 (2017)
16. M. Frandes, B. Timar, D. Lungeanu, A risk based neural network approach for predictive modeling of blood glucose dynamics. Stud. Health Technol. Inform. **228**, 577–581 (2016)
17. K. Fukushima, Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biol. Cybern. **36**(4), 193–202 (1980)
18. V. Gulshan, L. Peng, M. Coram, M.C. Stumpe, D. Wu, A. Narayanaswamy, S. Venugopalan, K. Widner, T. Madams, J. Cuadros, et al., Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. JAMA **316**(22), 2402–2410 (2016)
19. G.E. Hinton, Learning multiple layers of representation. Trends Cogn. Sci. **11**(10), 428 (2007)
20. G.E. Hinton, P. Dayan, B.J. Frey, R.M. Neal, The "wake-sleep" algorithm for unsupervised neural networks. Science **268**(5214), 1158 (1995)
21. G.E. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets. Neural Comput. **18**(7), 1527–1554 (2006)
22. G.E. Hinton, R.R. Salakhutdinov, Reducing the dimensionality of data with neural networks. Science **313**(5786), 504–507 (2006)

23. G. Huang, H. Lee, E. Learnedmiller, Learning hierarchical representations for face verification with convolutional deep belief networks, in *2012 IEEE Conference on Computer Vision and Pattern Recognition* (2012), pp. 2518–2525
24. A.G. Ivakhnenko, V.G. Lapa, Cybernetic predicting devices. Transdex (1966)
25. M.I. Jordan, T.M. Mitchell, Machine learning: trends, perspectives, and prospects. Science **349**(6245), 255–260 (2015)
26. K. Kamnitsas, C. Ledig, V.F.J. Newcombe, J.P. Simpson, A.D. Kane, D.K. Menon, D. Rueckert, B. Glocker, Efficient Multi-Scale 3D CNN with fully connected CRF for accurate brain lesion segmentation. Med. Image Anal. **36**, 61–78 (2017)
27. Y. Kim, J.W. Chong, K.H. Chon, J. Kim, Wavelet-based AR–SVM for health monitoring of smart structures. Smart Mater. Struct. **22**(1), 015003 (2012)
28. H. Larochelle, Y. Bengio, Classification using discriminative restricted Boltzmann machines, in *International Conference* (2008), pp. 536–543
29. Y. Lecun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L.D. Jackel, Backpropagation applied to handwritten zip code recognition. Neural Comput. **1**(4), 541–551 (2014)
30. T. Lee, S. Yoon, Boosted categorical restricted Boltzmann machine for computational prediction of splice junctions, in *International Conference on Machine Learning* (2015), pp. 2483–2492
31. M.K.K. Leung, H.Y. Xiong, L.J. Lee, B.J. Frey, Deep learning of the tissue-regulated splicing code. Bioinformatics **30**(12), i121–i129 (2014)
32. J.M. Levy, *Contemporary Urban Planning* (Taylor & Francis, London, 2016)
33. C. Li, X. Wang, W. Liu, L.J. Latecki, DeepMitosis: mitosis detection via deep detection, verification and segmentation networks. Med. Image Anal. (2018)
34. S. Liu, H. Zheng, Y. Feng, W. Li, Prostate cancer diagnosis using deep learning with 3D multiparametric MRI, in *Medical Imaging 2017: Computer-Aided Diagnosis*, vol. 10134 (International Society for Optics and Photonics, Bellingham, 2017), page 1013428
35. F.S. Lu, S. Hou, K. Baltrusaitis, M. Shah, J. Leskovec, R. Sosic, J. Hawkins, J. Brownstein, G. Conidi, J. Gunn, J. Gray, A. Zink, M. Santillana, Accurate influenza monitoring and forecasting using novel internet data streams: a case study in the Boston Metropolis. JMIR Public Health Surveill. **4**(1), e4 (2018)
36. J. Lyons, A. Dehzangi, R. Heffernan, A. Sharma, K. Paliwal, A. Sattar, Y. Zhou, Y. Yang, Predicting backbone cα angles and dihedrals from protein sequences by stacked sparse auto-encoder deep neural network. J. Comput. Chem. **35**(28), 2040–2046 (2014)
37. R. Miotto, L. Li, B.A. Kidd, J.T. Dudley, Deep patient: an unsupervised representation to predict the future of patients from the electronic health records. Sci. Rep. **6**(1), 26094 (2016)
38. M. Mohri, A. Rostamizadeh, A. Talwalkar, *Foundations of Machine Learning* (MIT Press, Cambridge, 2012)
39. A. Obinikpo, B. Kantarci, Big sensed data meets deep learning for smarter health care in smart cities. J. Sens. Actuator Netw. **6**(4), 22 (2017)
40. T. Pham, T. Tran, D. Phung, S. Venkatesh, DeepCare: a deep dynamic memory model for predictive medicine, in *Advances in Knowledge Discovery and Data Mining. PAKDD 2016*, ed. by J. Bailey, L. Khan, T. Washio, G. Dobbie, J. Huang, R. Wang. Lecture Notes in Computer Science, vol. 9652 (Springer, Cham, 2016)
41. S. Poslad, *Ubiquitous Computing: Smart Devices, Environments and Interactions* (Wiley, New York, 2011)
42. S. Rifai, Y. Bengio, Y. Dauphin, P. Vincent, A generative process for sampling contractive auto-encoders (2012). Preprint arXiv:1206.6434
43. F. Rosenblatt, The perceptron: a probabilistic model for information storage and organization in the brain. Psychol. Rev. **65**(6), 386 (1958)
44. R. Salakhutdinov, A. Mnih, G. Hinton, Restricted Boltzmann machines for collaborative filtering, in *International Conference on Machine Learning* (2007), pp. 791–798
45. J. Schmidhuber, Deep learning in neural networks: an overview. Neural Netw. **61**, 85–117 (2014)

46. J. Schmidhuber, Learning complex, extended sequences using the principle of history compression. Neural Comput. **4**(2), 234–242 (2014)
47. M. Shah, C. Rubadue, D. Suster, D. Wang, Deep learning assessment of tumor proliferation in breast cancer histological images (2016). Preprint arXiv:1610.03467
48. H.-C.C. Shin, M.R. Orton, D.J. Collins, S.J. Doran, M.O. Leach, Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4D patient data. IEEE Trans. Pattern Anal. Mach. Intell. **35**(8), 1930–1943 (2013)
49. P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. J. Mach. Learn. Res. **11**(Dec), 3371–3408 (2010)
50. B. Wei, Z. Han, X. He, Y. Yin, Deep learning model based breast cancer histopathological image classification, in *2017 IEEE 2nd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)* (IEEE, Piscataway, 2017), pp. 348–353
51. M.N. Wernick, Y. Yang, J.G. Brankov, G. Yourganov, S.C. Strother, Machine learning in medical imaging. IEEE Signal Process. Mag. **27**(4), 25–38 (2010)
52. S. Zhang, J. Zhou, H. Hu, H. Gong, L. Chen, C. Cheng, J. Zeng, A deep learning framework for modeling structural features of RNA-binding protein targets. Nucleic Acids Res. **44**(4), e32 (2015)