International Journal of Circuit, Computing and Networking

E-ISSN: 2707-5931
P-ISSN: 2707-5923
https://www.computerscience journals.com/ijccn
IJCCN 2024; 5(1): 37-46
Received: 25-01-2024
Accepted: 27-02-2024

Ali Rafea Salih AL-Shiblawi
Faculty of Technical and Engineering, South Tehran Branch, Islamic Azad University, Tehran, Iran

Corresponding Author:
Ali Rafea Salih AL-Shiblawi
Faculty of Technical and Engineering, South Tehran Branch, Islamic Azad University, Tehran, Iran

Deep learning-based user location estimation in SDN networks

Ali Rafea Salih AL-Shiblawi

DOI: https://doi.org/10.33545/27075923.2024.v5.i1a.67

Abstract
Indoor localization systems, which employ a variety of sensors on mobile phones, have advanced due to their popularity. Handover is needed to maintain communication when a mobile user travels between base stations (BS), particularly between cells. We presented and tested the CNN model for predicting SDN network users' upcoming positions using a portion of their actual mobility traces. We recommend incorporating convolutional neural networks to improve input data representation. The number of elements of each array antenna, the pencil parameter, the number of frequency reference points, and the distances between them based on the maximum delay were determined by dividing the urban telecommunication SDN network into 16 sub-sections and using 3 base stations. We also thought the mobile channel's core frequency was crucial to determining the user's whereabouts. Thus, we obtained 1,000 test samples for the requested data set. We developed deep learning architecture to estimate user locations in the next step. Using the Xception network architecture, Adam solver, and trainbr training function, we estimated user locations. Model implementation results show that the CNN model with 3700 repetitions assessed the user's location with 98.65% accuracy and 0.15 RMSE. R=0.91934 shows the results are associated.

Keywords: CNN, SDN, RSS, Machine Learning, Deep Learning

Introduction
Daily, millions of people utilize positioning systems and value them. Global positioning matters most. This gadget sends electromagnetic impulses in all directions to determine location. Its mistake is major. The electromagnetic wave spectrum demonstrates that these waves are too weak to be employed as infrastructure for a positioning system that can determine an individual's location in tight spaces owing to their distance. Smart mobile devices have enabled ubiquitous systems, and their demand is widely understood; thus, different infrastructures will be required to tackle this difficulty. Signal reception length, time difference, angle, and spatial fingerprint may be used to build such systems. Remember that some of these methods are inappropriate for intricate interiors. This study aims to reconcile SDN network internal and external location tracking. Popularity has improved indoor geolocation systems, which use cell phone sensors. WiFi-based indoor localization systems are innovative mobile phone location estimation methods. It can't be utilized everywhere since not all phones have Wi-Fi processors. Researchers offer localization methods using WiFi and mobile device inertial sensors like accelerometers, gyroscopes, and compasses. These sensors only function on high-end phones (Rizek *et al*., 2018) [31]. several people in industrialized and developing countries use low-end mobile phones without Wi-Fi, thus experts have offered several alternatives. Most mobile phones can receive cellular signals for geolocation, enabling it global localization. It also reduces energy use beyond phone use (Tian *et al*., 2015; Tiglao, 2021) [32, 33]. WiFi-based indoor localization is common. Smartphones' RSS from WiFi access points is used by WiFi-based indoor localization systems to create a user's WiFi location database. This uses biometrics. Fingerprinting is online and offline. Signal intensities are used to find known locations (signatures) in the target area in the offline phase. The fingerprint is used to develop a localization model to estimate the user's online location from the signal scan. Generate valid and unique signatures in space and on multiple devices or user behaviors is fingerprint-based approaches' main challenge (Rizek *et al*., 2021) [31]. However, SDN networks have garnered attention and advanced interior localization systems using device sensors. Wireless indoor localization systems estimate network users' locations. Because not all networks have Wi-Fi processors, the technology cannot be used everywhere.

Localization systems using WiFi and inertial sensors like accelerometers, gyroscopes, and compasses have been proposed (Rizek *et al*., 2018) [31]. Since many people in developed and developing countries use low-end wireless services and technologies that don't support Wi-Fi, researchers have recommended many ways. This applies to these networks, as proved by Tian *et al*. (2015) [32] and Tiglao (2021) [33]. WiFi-based indoor localization is common. WiFi access point RSS locates users and creates a WiFi location database in WiFi-based indoor localization systems (Rizek *et al*., 2021) [31]. AI is advancing quickly, from self-driving vehicles to SIRI. Scientific literature depicts AI as human-like automata, yet it might be anything. 2018 (Javandia) AI application machine learning (ML) can autonomously compute sophisticated mathematical calculations on enormous datasets. Digital data is used to develop generic models using machine learning. The model predicts and extracts information (Baghbani, 2016) [1]. Building an autonomous computer system that improves with experience is shown in ML. (2015) Jordan *et al*. The evaluation of actions and functions by machine learning is similar to experience. ML lets people find structures and predict using enormous data sets. ML requires good learning, plenty of data, and dependable compute. Tsai *et al*. (2015) [40] ML is promising for large dataset analysis.

2. Theoretical foundations

We understand positioning system relevance. The second and third generations of mobile users have restricted data transfer speeds and are geared for voice and video communication. The third generation of laptops featured GPS since densely populated or limited regions may make position information hard to get (Zhou *et al*., 2015) [34]. Commercial GPS accuracy is 10 meters. GPS needs a direct line of sight to numerous satellites, making it unsuitable for indoor usage. Additionally, GPS devices are expensive (Ali *et al*., 2019) [6]. GPS errors are common in noisy, busy areas. So, fourth-iteration system study started with positioning system definition. Any-time, exact positioning system. The fourth generation should solve coverage, data transfer rate, and power usage. In addition to revolutionary aerial and array antenna approaches, wireless network architecture must be fundamentally changed to achieve this goal. Using the cellular network paradigm with point-to-point communication, which is exclusively utilized in ad hoc networks, improves network design (Malik *et al*., 2019) [5]. The chapter begins with cellular communication networks. A quick introduction to several of the most popular mobile user location methods is offered, along with a basic discussion of their methodology. The localization method based on signal strength is then presented and discussed. Since mobile cellular communication networks are the system paradigm, their structure will be addressed first. Mobile data transmission to the public telephone network is improved by the fixed telephone network (Zhou *et al*., 2015) [34]. Because mobile communications have limited frequency resources, frequency reuse is common. If suitably spaced, several BSs may utilize the same radio channel without interference. If N cells employ S frequency channels, k=S/N calculates the number of channels in each cell. BSs may use sectorized directional antennas to decrease interference. The graphic shows sectorization. Cellular structure boosts system capacity and reduces electricity needed to convey data. Network-based and mobile-based positioning algorithms exist. Due to the network's processing power, network-based algorithms may apply more advanced algorithms for positioning calculations.

Each positioning method has two steps, independent of calculation location. In the first step, signal propagation time, reception angle, and power are measured relative to known reference sites. The location is approximated in the second step using data and processing methods. Mobile users are identified (Gu *et al*., 2019; Gezichi, 2008) [19, 8]. The positioning approach uses signal arrival time to compute signal propagation time between the reference location and the mobile user and signal propagation speed to calculate distance. The intersection of the circles produced from the mobile user's geometric location determines their position given this distance. Unfortunately, this technique requires perfect BS clock pulse-mobile user synchronization (Gu *et al*., 2019; Gezichi, 2008) [19, 8].

Positioning measures signal propagation time between two reference places and the mobile device using signal arrival time. TDOA is better than TOA since it does not need transmitter-receiver synchronization. The hyperbolic geometric location in this method necessitates more complex equations than the TOA method (Roxin *et al*., 2007; Elgamodi *et al*., 2021) [9, 10]. The positioning technique measures signal reception angle using the signal entry angle as a reference. AOA may be approximated using array antennas in the network structure, although accuracy depends on antenna layout and LOS circumstances. Antenna arrays increase system complexity (Roxin *et al*., 2007; Elgamodi, 2021) [9, 10]. The mobile user measures the power of the received signal from reference points and processes the power information to determine their position because the received power depends on their position. Multiple signal reception routes produce reflection, dispersion, and fragmentation. Modeling small-scale fading-induced signal amplitude changes using a Rayleigh distribution function (Roxin *et al*., 2007; Elgamodi, 2021) [9, 10]. RSS has two placement mechanisms. The first solution is geometric and accounts for MS-BS distance's effect on signal power loss. Using the signal propagation loss model, the RSS value relative to a BS is understood as the MS-BS distance, and the MS location is determined using a geometric technique that accounts for numerous BSs. At least three reference points are needed to accurately position a mobile user in two-dimensional coordinates and four in three-dimensional coordinates (Roxin and *et al*., 2007; Algamodi, 2021) [9, 10]. Data is stored in a database. In the second step, the positioning phase or online phase, the MS compares the signal strength from the BSs to the RSS values in the database and uses a search algorithm to identify its location. Presently (Roxin *et al*., 2007) [9]. To enhance fingerprint location accuracy, several calibration tests are needed; interpolation methods are employed to minimize the number. This approach has many downsides. First, improving positioning precision increases the calibration stage's complexity. Second, nearby environmental changes affect RSS reception. Indeed, environmental changes invalidate database data, reducing location accuracy. Overall, RSS-based approaches are cheaper and easier. Many wireless systems measure RSS. RSS systems are less accurate than TOA systems, which is why they are seldom employed in commercial applications (Figueiras, 2008) [11]. Every RSS-based route attenuation research assumes the channel is free before system usage.

Average path attenuation, shadow effect, and small-scale fading affect mobile user power. RSS-based positioning is more popular because to the cost-precision trade-off at this method, reference nodes (RN) are purposefully put at known places, and the sensor analyzes signal intensity between RN and points with unknown position (BN). Due to obstacles, signal attenuation varies, making this approach less exact than TOA, TDOA, and AOA (Shlabi *et al*., 2019) [12]. Machine learning models are built via training and testing. The pattern of training data is used to modify model parameters. Using test data, the trained model's learning rate is assessed. Multiple elements must be examined to assess a machine learning model. Machine learning models are assessed by test data correctness, processing speed, and model volume. A system learns a function from input to output using input-output pairs in supervised learning. System training using supervised learning needs input data. However, certain issues cannot be solved by supervised learning systems.

These challenges are difficult to tackle using supervised learning. Reinforcement learning models such issues. It is a sequential optimization issue for dynamic programming (DP) called approximation dynamic programming. The general reinforcement learning model involves a controlled agent with a rhythmic template monitoring the system state at each time step t and receiving a reward rt from its environment/system after performing an action. A policy-mandated action moves the system to state st+1. Every interaction improves the reinforcement learning agent's environment knowledge. Yilmaz and Haydari, 2020. Unsupervised learning: Support vector machine associative rules are the most successful and extensively used unsupervised learning technique, using 15%. For classifier data classification, decision tree (DT) classifiers are the most used. Decision trees are powerful methods used in pattern identification, image processing, and machine learning.

A numerical characteristic is consistently and efficiently compared to a threshold value in each baseline test in DT. Constructing conceptual conceptions is much simpler than calculating neural network node weights. DT is mostly used for categorization. DT is a common data extraction categorization model. This paradigm's trees have nodes and limbs. Nodes indicate categorization category properties, whereas subsets specify potential values. Decision trees are popular for their simple, reliable processing of many data kinds. The Iterative Dichotomy 3 (ID3), its successor (C4.5), the Classification and Regression Tree (CART), the CHi Square Automatic Interaction Detector (CHAID), the Multivariate adaptive regression lines (MARS), the generalized, unbiased, interaction detection and estimation (GUIDE), and the conditional inference trees (CTREE) are the most important decision tree algorithms. The most important criterion in measuring the performance of decision trees is the entropy criterion. Entropy is used to measure the impurity or randomness of a data set. The entropy value is always between 0 and 1. Its value is better when it is equal to 0, while it is worse when it is equal to 1, that is, the closer its value is to 0, the better. This index is calculated based on the following relationship:

$$Entropy(S) = \sum_{i=1}^{c} P_i \log 2^{P_i}$$

In this regard, P_i is equal to the rate of the number of subset samples and the i-th characteristic value.

- Artificial neural networks (ANN) have been a hot topic in AI since the 1980s. A simple model and connected networks are built to simulate the brain's neural network. This is information processing. Neural networks are computer models with numerous neurons. Every activation function network node represents a separate output function. Connections between nodes represent signal weights, which are the network's memory. Network output depends on connection, weight, and incentive function. The network usually approximates a natural algorithm or function or a logical approach (Wu *et al*., 2018) [36].
- Components of a Neural Network

1. Inputs

The input signals are equivalent to the input neural signals and they form the input of the neuron, which can be the output of other layers. The inputs can be as follows:

- Numerical data.
- Literary, technical and other texts.
- Image or shape.
- Weight vector

The weight vector is equivalent to the synaptic information values of neuron inputs. The influence of the input $x_i$on the output $y_i$is measured by the weight characteristic. W_i is adjustable and adjusted based on the transformation function and the type of learning algorithm. The sum function processes neuron data. In solitary neuron networks, the output of the problem is determined in part by the sum function. In multineuron networks, the activity level of neuron j in the interior layers is determined by the sum function. The addition function is shown in the following relation. In this relation, *Xj* are equivalent to network inputs and W_{ij} are equivalent to network weight vector.

$$\sum x_j w_{ij}$$

The activity function determines the output values of a synthetic neuron based on its input values. The activity function takes into account a threshold for the output values of each neuron. The activation rule determines how a neuron responds to each input pattern with an activation response. The activation function converts a broad range of input values to a single output value.

The output means the answer to the problem. The output of the neuron is determined from the above relationship. In this relationship, x_j and $w_{ij}$are the values of the following relationship.

$$Y_i = \text{Activation Function}\left(\sum x_j w_{ij}\right)$$

Neural Network Capabilities

- Calculation of a known function
- Approximation of an unknown function
- Pattern recognition
- Signal processing

- Learning to do the above (Wu *et al*., 2018) [36].

Spoofnet-dcn-1: There are four hidden layers in this network. Convolutional sublayers with either maximum or average integration make up the first three hidden layers. The two sublayers are fully and internally joined in the fourth layer. For low- and high-traffic data categories, the output layer is a 2-unit softmax layer; for prediction, linear support vector machines (SVMs) are employed. The number of neurons in the network's input layer, 32832 (or 64664), represents the volume of input data. The initial hidden layer is a convolutional layer employing local acceptance fields of size 575 (or 975). There are 64 feature maps in total. The integration is applied to this layer and subsequent layers in 373 regions. Convolutional layers with a 5 x 5 (or 7 x 7) local reception field, 64 feature maps, and 64 input channels are the second and third concealed layers. There are 150 neurons in an internal, fully interconnected sub-layer for the fourth layer's multiplication, which are all interconnected. The output of the layer is two units from the softmax layer.

Spoofnet-dcn-2: Three levels comprise the envisioned network. It is found in maximal integration, normalization, convolutional sublayers, and each of the three hidden layers. that the third layer below the typical layer does not contain. This layer produces 32 feature maps as its output. The first concealed layer is a convolutional sublayer with a maximum integration step and also normalization using local acceptability fields of size 575 (or 975). In 3x3 regions, the merge sublayer is applied to all layers. There is a convolutional sub-layer with a mean and normal integration phase, a 5x5 (or 7x7) local reception field, 32 input channels, and 32 output feature maps in the second hidden layer. Next, in the third layer, the convolutional and integration sub-layer transforms the 32 input channels into 64 feature maps, and at the network's output is a softmax layer with two categories, low traffic and high traffic.
In lieu of sigmoid or... activation functions, we employ the corrected linear unit (ReLU), which results in a considerable increase in training speed (Krizhevsky *et al*., 2017) [40].

- **Deep convolutional network training**

General process
The learning method has two training control stages. All training samples must be entered into the network to complete a cycle. The network iterates after weight updates. The trained network's prediction performance is assessed after each validation set cycle. Separating the validation and training sets allows generalization assessment. Training and assessment continue until the validation set's predictive power peaks. Azizpour and colleagues (2015) propose continuing activities until the network is unsuitable.
Since repetitive experiments are required to arrive at a suitable network architecture, it is crucial to have a distinct third data set to execute the final evaluation.
Meta parameters affect neural network performance prediction and must be specified before learning. Layer count, kernel size, and gradient descent parameters are included. Only empirical data and experience inform their structure. The instructional method's regulating and guiding factors are highlighted below. The meta-parameter weights were updated using this equation:

$$w \leftarrow w' = w - v$$

$$v \leftarrow v' = \gamma v + \frac{\eta \partial C}{\partial w} + \frac{\lambda}{2n}\sum_i w_i^2$$

Learning rate η: This parameter determines the weight update, which influences the network's convergence rate directly. A large E can cause problems by bypassing the optimal solution, while a small E can result in a prolonged training period.

Momentum γϵ(0,1): This parameter specifies the inertia of gradient update, or the weight assigned to the most recent gradient. Large la reduces the SGD's volatility.

Verne's attenuation ϵλ(0,1]: The weight adjustment is defined by this parameter. Furthermore, it is imperative to gradually decrease the pace of learning. This can be achieved through a predetermined factor reduction over multiple periods or exponential attenuation. Within the exponential attenuation clause, the parameter A determines the learning rate attenuation speed for the number of repetitions k:

$$\eta \leftarrow \eta' = \frac{\eta}{1+kv}$$

Reduce excess fit
By adding a second term (equation), the introduced cross-entropy cost function can be improved. This second element is known as the L2 norm, a weight adjustment.

$$C = C_0 + \frac{\lambda}{2n}\sum_i w_i^2$$

Here, multiple addition operations are performed on each weight. We can be an arbitrary cost function, but in our case $C_0$is the cross-entropy. The second term is defined as the sum of the squares of the weights of a neuron with the scaling factor $\lambda/(2n)$, where $\lambda > 0$ indicates the importance of the second term and n defines the number of input samples. There is no predefined optimal value for λ therefore its value must be determined through investigation. As a consequence of global minimization, the regularization statement is invoked in order to minimize all input weights. Consequently, the network favors learning small weights and will only permit large weights when the cost of the first term decreases. The λ parameter plays an important role in determining the preference of learning small weights over C_0 minimization.
By adding variation to sample data, data augmentation gains fake data. Any label-preserving change of sample data is allowed if the variation generated is meaningful. Natural pictures need translation, rotation, scaling, and reflection-resistant objects. Many of these transformations, especially rotation and reflection, are not naturally invariant to ConvNets, hence they seldom learn them showed that ConvNet transition, scaling, and deformation decrease with layer count. Transformations that incorporate significant variation in the training data drive the network to learn invariant features, improving generalization and lowering overfitting. Common approaches have been used, but more elaborate patterns are available. Additional data has a big impact. Data augmentation is usually requested during training. A random selection determines whether and what

to modify the sample. It will prevent the network from receiving the same data. Roth *et al*. (2015) [39] include random elimination as the third overfitting reduction method. During training, random deletion removes neurons and their connections from a network using the Bernoulli distribution. Normal probability is 0.5. In actuality, neuronal output is zero to randomly kill neurons. This stochastic mechanism synchronizes neurons so they cannot depend on a certain collection of active or existing neurons. Each neuron must strengthen its connections with other input neurons to build more robust feature representations. Random elimination cannot test or evaluate, preventing the network from predicting at full capacity. The findings show significant overfitting reduction. Krizhevsky *et al*. (2017) [40] used 0.50-probability random deletion in two of the final three completely linked layers. Overfitting decreased as network training time increased.

Spoofnet-dbn-1: In this architecture, there is a structure as 256-256-2-512-1024 (or 4096-512-256-256-2), where the value of 1024 (or 4096) corresponds to the dimensions of the data The input is 32732 (or 64664) and the value 2 in the output corresponds to the number of categories, i.e. the two categories of low-traffic and high-traffic data. In the first and second hidden layers, there are 512 and 256 variables, respectively, which are trained generatively. The third hidden layer also contains 256 variables, which are separately trained along with the labels of the categories. Each concealed layer is trained avidly.

Spoofnet-dbn-2: In this architecture, the input and output dimensions are the same as the preceding structure, except that the number of variables in each finite Boltzmann machine is distinct. The first, second, and third concealed layers have 256, 128, and 64 variables, respectively. The first two layers are trained generatively and the third layer is separable.

Training deep belief networks

The greedy learning method for deep belief networks is based on a basic premise. First, it trains the lower bound Boltzmann machine with W^1 parameters. Then the weights of the second layer are initialized with $W^2 = W^{1T}$ to verify that the deep belief network's hidden layers are at least as excellent as the initial bounded Boltzmann machine. Following the extraction of the values of layer h4, the deep belief network can be improved by using this data and modifying W^2 Of course, in general, there is no need to make the size of the weight matrix W^1 and W^2 the same.

This idea can be used to train the third layer of the restricted Boltzmann machine on the vector h^2 obtained from the second restricted Boltzmann machine. By setting $W^3 = W^{2T}$, we guarantee that we improve the lower bound of the logarithm of the exponential. We can also improve the bound by changing W^3. This layer-by-layer process may be performed several times to produce a deep hierarchical model. The following algorithm demonstrates this technique.

One of the most well-known techniques for optimizing predicted solutions is Support Vector Machine (SVM). SVM was developed by Vapnik as a kernel-based machine learning model for regression and classification applications. SVM's outstanding generalizability, optimal solution, and discrimination capability have drawn the attention of the data mining, pattern recognition, and machine learning communities in recent years. SVM has been utilized as a potent instrument to tackle practical binary classification problems.

In the SVM method, we assume that we have the set of data points $\{(x_1, c_1), (x_2, c_2), \ldots, (x_n. c_n)\}$ and we want to divide them into two classes $c_i = \{-1, 1\}$ separate. Each $x_i$is a p-dimensional vector of real numbers, which are essentially the same variables describing the software's behavior. Methods of linear classification attempt to separate data by constructing a hypersurface (a linear equation). The support vector machine classification technique, which is one of the linear classification methods, determines the optimal hypersurface that maximally separates the data of two classes. In order to facilitate comprehension of the topic,

The main problem is to minimize $\frac{1}{2}\|w\|^2$considering the constraint 1-yi(wTxi + b) ≤ 0 for $i = 1, \ldots, n$. In this case, the Lagrangian function is:

$$l = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \sum_{i=1}^{n} \alpha_i(1 - y_i(\mathbf{w}^T\mathbf{x}_i + b))$$

It should be noted that ||w||2 = wTw. By deriving from l we will have:

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \,, \rightarrow w = \sum_{i=1}^{n} \alpha_i y_i x_i \quad w + \sum_{i=1}^{n} \alpha_i(-y_i)x_i = 0$$

Support vector machines multidimensionalize input vectors. Building a hypersurface will separate input vectors by maximum distance. This is the "super surface with the maximum separating boundary" hypersurface.

One of the non-parametric regression functions is the nearest neighbor method. If a data series $D_N = \{\{y_i, x_i\} \in R_+^1 \times R_+^l, i = 1, \ldots, N\}$ is available, a non-parametric regression estimator, y^{NP}, will fit the following cost function to the time series vector DN (training data) by minimizing its length:

$$C(y^{NP}) = \frac{\sum_{i=1}^{N}(y_i - y^{NP})^2 K(x_i, x_0, x_l)}{N}$$

In this regard, the output variable is represented by $y_i$and the input variables are represented by the vector x_i. The input variables may be precipitation, flow, flow with varying delays, or any other variable, resulting in a combination of these variables at one or more stations. $x_0$specifies the question point and is selected from the test data set in the input space to be The output variable values are calculated according to x_0. Similar to D_N, another set is considered as test data or D_M, so that these data do not have any members in common with the data related to the training section or D_N, and the question point is also selected from this data (Test data set) and corresponds to the variable It is independent of X vector. Also, the dependent variable is used in the test data set to test the estimator

according to the data information of the training section or D_N. The K function also represents the weight or kernel function, the value of which is based on the Euclidean distance between each point in the training section and the query point in the test section, as calculated by the following equation:

$$K(x_i, x_0, b) = \{\begin{matrix} 1 \ if \ ||x_i - x_0|| \leq b \\ 0 \ if \ |x_i - x_0| > b \end{matrix}\}$$

It represents both the Euclidean distance and the neighbourhood radius. Minimization is done according to the value of the parameter y^{NP} and the kernel defined in the above relation. For more estimation, the following relationship is used:

$$y_{pred}^{nn} = \frac{\sum_{i \in I_{nn}} y_i}{|I_{nn}|}$$

Where, I_{nn} is a set whose members, its index is the number of observation data inside the circle. E are located at the radius b from the question point and $|I_{nn}|$ The number of members of the set I_{nn} is:

$$|I_{nn}| = \{i: ||x_i - x_0|| \leq b\}$$

This method's performance efficacy is dependent on the selection of parameters b (neighborhood radius) and 1 (number of delays in input parameters).

The self-organizing network of Cohen is an unsupervised neural network that seeks to reduce dimensionality and aggregate data. Kuhnen is distinct from other neural networks because it uses the neighborhood function to conserve the spatial characteristics of the input space. Each Kohen network is comprised of multiple nodes (trons). Each node has a weight vector. This vector's dimensions are identical to those of the input space. After training the network, each region of network nodes responds to particular input data patterns. Competitive learning network is the training technique. When a new training sample is applied to the network, its Euclidean distance is computed using the weight vectors of each node. The winning node is the one whose weight vector is most similar to the input vector. The weights of every neuron are modified using the following formula.

$$W_i(t+1) = W_i(t) + \theta(v,t)\alpha(t)(X(t) - W_i(t))$$

Where, $\alpha(t)$ is the decreasing learning rate and $X(t)$ is the input vector. The neighborhood function depends on $\theta(v,t)$, depending on the distance between neuron i and BMU in the network. In addition, the neighborhood radius and learning rate decrease over time according to the following equations.

$$\alpha(t) = a_0 \exp\left(-\frac{t}{\lambda}\right), \quad t = 1,2,3, \ldots.$$

Where t the repetition is number and λ is the constant value and a_0 is the initial learning rate.

$$\theta(t) = \exp\left(-\frac{dist^2}{2\sigma_{(t)}^2}\right)$$

That $dist$ is the distance from BMU and $\sigma(t)$ is calculated using the following formula.

$$\sigma(t) = \sigma_0 \exp\left(-\frac{t}{\lambda}\right), \quad t = (1,2,3, \ldots.)$$

In this formula, σ_0 is the width of the network at time t_0, λ is a constant value, and t is the current iteration number.

1982's Hopfield network is an associative neural network (Hopfield, 1982). Hopfield and Tank developed and applied this network to solve optimization problems in 1985 (Hopfield *et al*., 1985).

Hopfield network training algorithm

Using the following algorithm, Hopfield network optimization was used to identify the optimal solution to the high cost function:

In recurrent neural networks, input and state series are present. These states are the output of the network at any given time, and in addition to the inputs, outputs, or states of the preceding stages, they contribute to the production of the output at any given time. Moreover, both the state and the input are functions of time, which are denoted by x_t and h_t. In these networks, the output of each stage is returned to the network, combined with the new input, and then used to generate the output of the subsequent stage. This procedure is depicted by the following relationship.

$$h_t = f_w(h_{t-1}, x_t)$$

In the above relation, h_t is the output or the new state and it is actually a function of the previous state h_{t-1} and also the new input x_t. And in addition, f_w is a non-linear function consisting of network weights, and the $tanh$ function is usually used. As a result, the above relationship can be rewritten as follows.

$$h_t = tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

In the above relation, the weighted combination of h_{t-1} and x_t is passed through the tanh function. Consequently, the network has two distinct weights, one related to the previous state and the other related to the input.

In general, the working procedure in recurrent neural networks is such that at the zero moment, we have the input Wh, which is a part of the input. This input x_0 enters the network and produces the state h_0. This state h_0 re-enters the network and is combined with the new input x_1 and then passed through a $tanh$ function and finally produces the output h_1. Therefore, this h_1 is dependent on both x_1 and h_0, and then in the next step, the same h_1 enters the network and by combining this h_1 with the input x_2, the h_2 state is produced. In fact, this h_2 is dependent on h_1, and

this same h_1 is dependent on h_0. Therefore, it can be said that each of these h states is dependent on the previous states with the weight determined for them, and the same process continues until the end.

3. Research Method

Using RSS, distance-based methods estimate signal intensity. This approach calculates the distance between each device (user) and the target using the target's average power and the signal transmission channel's attenuation with distance. Each device's distance from the target is combined in the control centre to estimate the target's position. Energy collected at each place determines the radius of circles utilized to meet device foci in two-dimensional surroundings. One reason target location is inaccurate is the bases' nondirectionality. However, directed devices may provide more about the target's unknown location.

The main results of using the deep learning method to determine a user's location within an SDN network will be discussed in this chapter. Based on this, we will first describe the CNN algorithm settings and outcomes, after introducing the database that transmits the factors impacting the location estimate. It should be made clear that the simulation in this study is conducted on a system equipped with an 8 GB RAM and a 5-core CPU using MATLAB version 2021a.

- Sorting the data: We must organize the data so that the MATLAB software can comprehend and read it. In this study, the number of rows represents the number of user position-related states, while the number of columns represents the factors influencing the estimation of user position.
- Labeling of data: In addition, since MATLAB software solves numerically, all variables must be labelled numerically. All parameters and targets in this section are designated in Latin.
- Normalize the data: Now that they have all been converted into numeric variables, the data can be readily loaded into MATLAB. In addition, it is important to note that, because the data have various scales, they must be converted into a standard format. Standard format requires placing all data between d1 and d2 using the following formula:

$$\bar{x} = \frac{(x-xmin)(d2-d1)}{xmax-xmin} + d1$$

According to the data, d1=0 and d2=+1 have been selected. Next, we save all the data with the specified specifications and labels in a file called "User_L.xls"

In this study, 70% of the data is used to train deep neural networks and 30% is used to validate the model. These divisions are entirely discretionary in order to utilize all data for both objectives. The Randperm function of the MATLAB software can automatically generate a random index, and the associated data is stored in the corresponding matrices. 80% of the dataset, or 800 rows, are used to train the classification model, while 20% of the original dataset, or 200 rows, are used to evaluate the model. is utilized.

The results will be presented within this section. Consequently, Figure (1) depicts the layered architecture of the network.

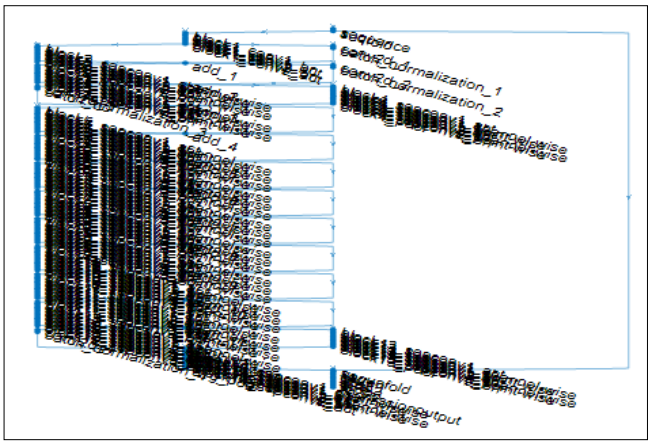

Fig 1: Deep network layer architecture

Figure (2) depicts the deep model convergence diagram that led to the estimation of the user's position.

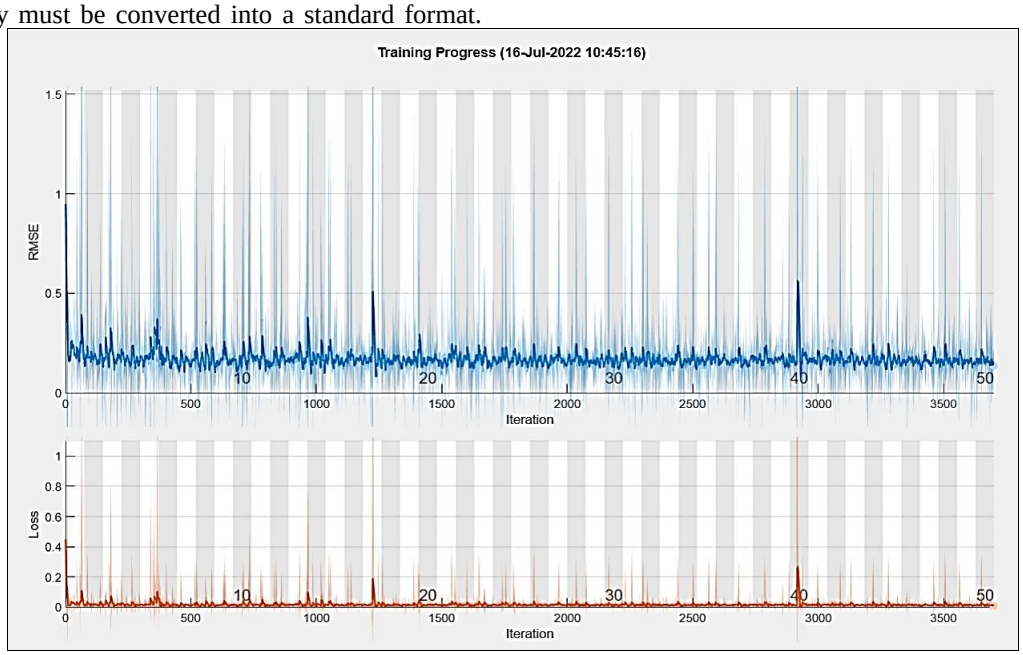

Fig 2: The suggested CNN model's convergence diagram, which has an accuracy of 98.65%

As depicted in Figure (2), the CNN model with 3700 iterations has an average RMSE error of 0.15 and an accuracy rate of 98.65 percent when estimating the position of users.

In addition, Figure (3) depicts the proposed CNN network's correlation diagram based on the distribution of training and testing data. As can be seen, the correlation coefficient between the results is R=0.91934.

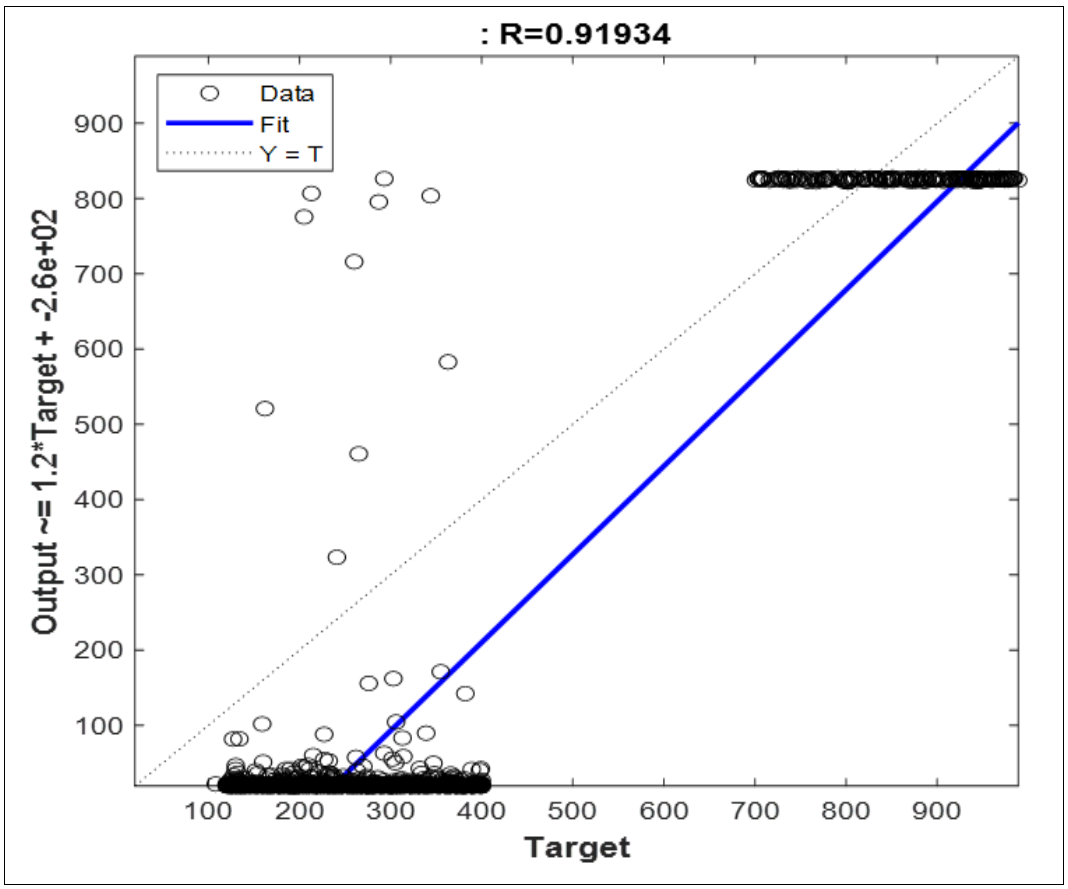

Fig 3: Data correlation diagram

Figure (4) also shows users' anticipated position based on deep learning and observed data, as well as their current location. As is evident, deep learning has produced precise user location estimates.

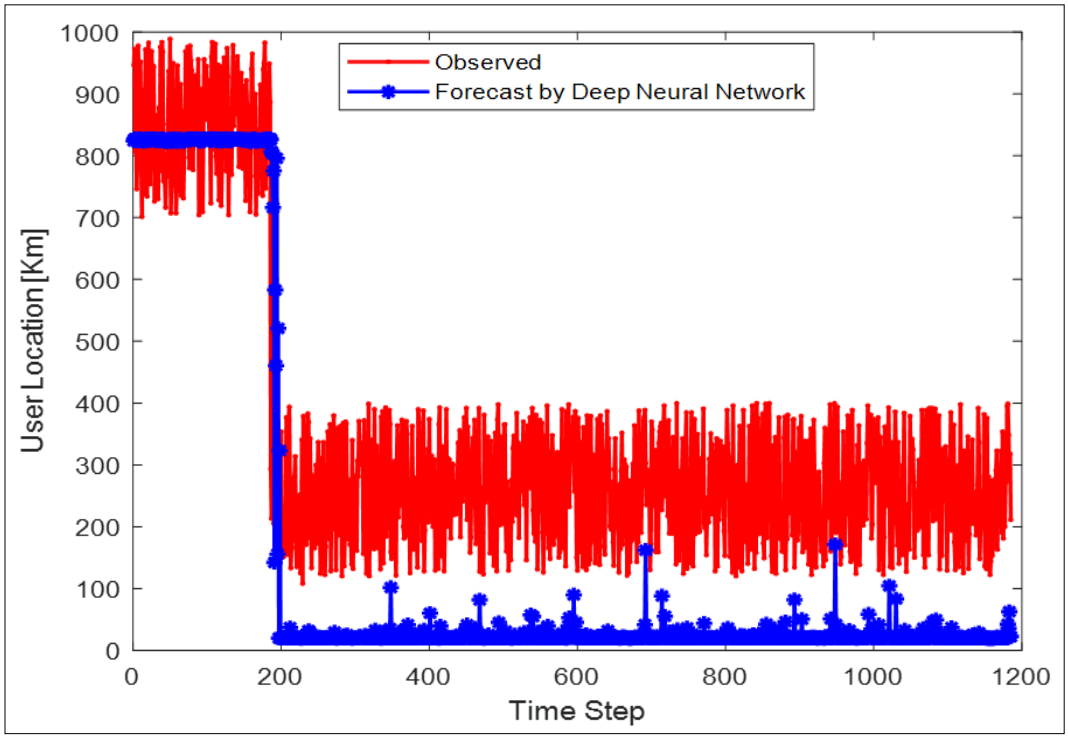

Fig 4: Arbaran position estimation based on observed data

Figure (5) illustrates how the CNN model can estimate the future position of users based on their context in the network.

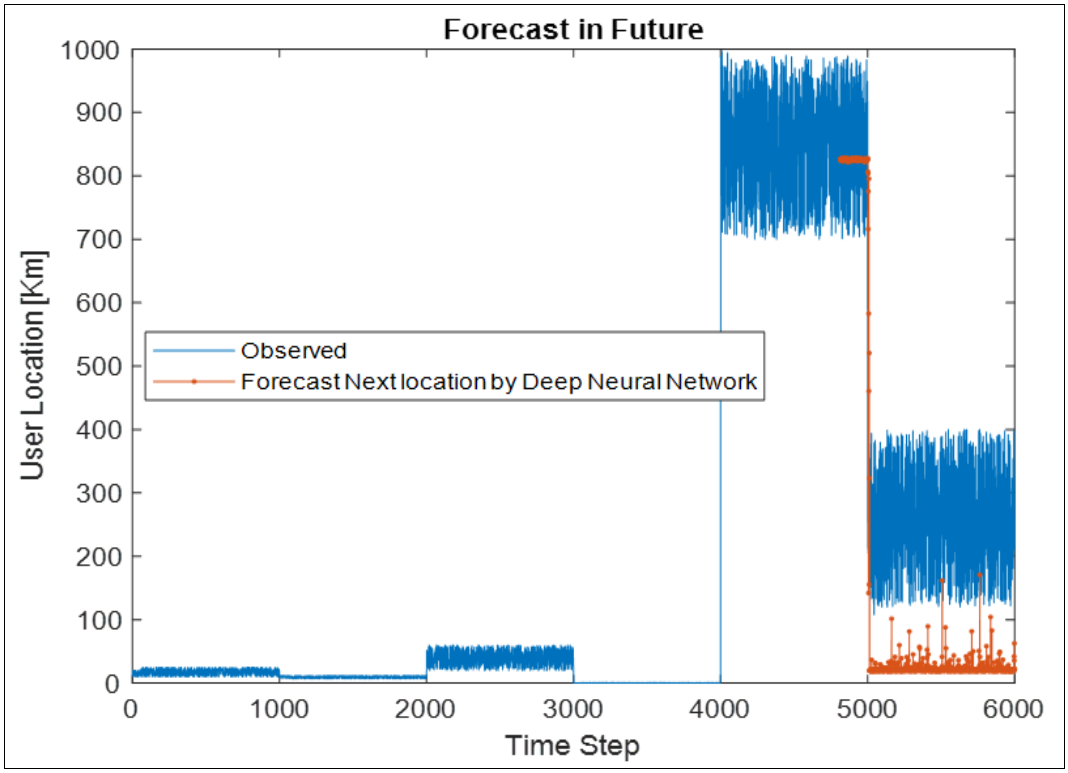

Fig 5: CNN is used to estimate users' future positions.

5 Conclusions

We believe the mobile channel's core frequency affects the user's location. Thus, 1,000 test samples were acquired for the specified data set. Next, we apply deep learning to estimate user location. Using the Exception network architecture, Adam solver, and trainer training function, we estimated user locations. Model implementation results show that the CNN model with 3700 repetitions assessed the user's location with 98.65% accuracy and 0.15 RMSE. R=0.91934 shows the results are associated.

References

1. Baghbani S. Techniques and methods of machine learning on big data. National Conference of New Technologies in Electrical and Computer Engineering; c2016.
2. Mohammadi MA, Sepidnam Q, Arad Y. Estimation of internal location in Wi-Fi network based on post-error propagation neural network. 2nd National Conference of Knowledge and Technology of Electrical, Computer and Mechanical Engineering of Iran, Tehran; c2017.
3. Keshavarz H, Mirsalehi M, Zamiri H. Estimation of the application location of mobile communication using multi-layer perceptron network. 11th Electrical Engineering Conference, Shiraz; c2013. Available from: https://civilica.com/doc/152237.
4. Zhu J, Durgin GD. Indoor/outdoor location of cellular handsets based on received signal strength. In: 2015 IEEE 61st Vehicular Technology Conference. IEEE. 2015;1:92-96.
5. Malik H, Khan SZ, Sarmiento JLR, Kuusik A, Alam MM, Le Moullec Y. NB-IoT network field trial: Indoor, outdoor and underground coverage campaign. In: 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC. IEEE; c2019. p. 537-542.
6. Ali WH, Kareem AA, Jasim M. Survey on wireless indoor positioning systems. Cihan University-Erbil Scientific Journal. 2019;3(2):42-47.
7. Guo X, Ansari N, Hu F, Shao Y, Elikplim NR, Li L. A survey on fusion-based indoor positioning. IEEE Communications Surveys & Tutorials. 2019;22(1):566-594.
8. Gezici S. A survey on wireless position estimation. Wireless Personal Communications. 2008;44(3):263-282.
9. Roxin A, Gaber J, Wack M, Nait-Sidi-Moh A. Survey of wireless geolocation techniques. In: 2007 IEEE Globecom Workshops. IEEE; c2007. p. 1-9.
10. Elgamoudi A, Benzerrouk H, Elango GA, Landry R. A survey for recent techniques and algorithms of geolocation and target tracking in wireless and satellite systems. Applied Sciences. 2021;11(13):6079.
11. Figueiras J. Accuracy enhancements for positioning of mobile devices in wireless communication networks. Department of Electronic Systems, Aalborg University; c2008.
12. Shalaby M, Shokair M, Messiha NW. Performance of RSS based cooperative localization in millimeter wave wireless sensor networks. Wireless Personal Communications. 2019;109(3):1955-1970.
13. Doan A, Madhavan J, Domingos P, Halevy A. Ontology matching: A machine learning approach. In: Handbook on Ontologies. Springer, Berlin, Heidelberg; c2004. p. 385-403.
14. Liu X, Athanasiou CE, Padture NP, Sheldon BW, Gao H. A machine learning approach to fracture mechanics problems. Acta Materialia. 2020;190:105-112.

15. Xiong J, Shi SQ, Zhang TY. A machine-learning approach to predicting and understanding the properties of amorphous metallic alloys. Materials & Design. 2020;187:108378.
16. Islam MM, Nasrin T, Walther BA, Wu CC, Yang HC, Li YC. Prediction of sepsis patients using machine learning approach: A meta-analysis. Computer Methods and Programs in Biomedicine. 2019;170:1-9.
17. Straub J. Machine learning performance validation and training using a 'perfect' expert system. MethodsX. 2021;101477.
18. Geysen D, De Somer O, Johansson C, Brage J, Vanhoudt D. Operational thermal load forecasting in district heating networks using machine learning and expert advice. Energy and Buildings. 2018;162:144-153.
19. Gu W, Foster K, Shang J, Wei L. A game-predicting expert system using big data and machine learning. Expert Systems with Applications. 2019;130:293-305.
20. Livieris IE, Drakopoulou K, Tampakas VT, Mikropoulos TA, Pintelas P. Predicting secondary school students' performance utilizing a semi-supervised learning approach. Journal of Educational Computing Research. 2019;57(2):448-470.
21. Bowyer C, Greene D, Ward T, Menendez M, Shea J, Wong T. Reinforcement learning for mixed cooperative/competitive dynamic spectrum access. In: 2019 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN). IEEE; c2019. p. 1-6.
22. Mansour AM. Decision tree-based expert system for adverse drug reaction detection using fuzzy logic and genetic algorithm. International Journal of Advanced Computer Research. 2018;8(36):110-128.
23. Bui DK, Nguyen T, Chou JS, Nguyen-Xuan H, Ngo TD. A modified firefly algorithm-artificial neural network expert system for predicting compressive and tensile strength of high-performance concrete. Construction and Building Materials. 2018;180:320-333.
24. Smisek R, Hejc J, Ronzhina M, Nemcova A, Marsanova L, Kolarova J, Vitek M. Multi-stage SVM approach for cardiac arrhythmias detection in short single-lead ECG recorded by a wearable device. Physiological Measurement. 2018;39(9):094003.
25. Li J, Cheng JH, Shi JY, Huang F. Brief introduction of back propagation (BP) neural network algorithm and its improvement. In: Advances in Computer Science and Information Engineering. Springer, Berlin, Heidelberg; c2012. p. 553-558.
26. Fletcher R, Reeves CM. Function minimization by conjugate gradients. The Computer Journal. 1964;7(2):149-154.
27. Fahmy I, Ayman S, Rizk H, Youssef M. MonoFi: Efficient indoor localization based on single radio source and minimal fingerprinting. In: Proceedings of the 29th International Conference on Advances in Geographic Information Systems; c2021. p. 674-675.
28. Wainer G. RSSI-based indoor localization with LTE-A ultra-dense networks. In: 2020 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS). IEEE; c2020. p. 1-7.
29. Alhomayani F. Deep learning methods for fingerprint-based indoor and outdoor positioning [Doctoral dissertation]. University of Denver; c2021.
30. Alawami MA, Kim H. LocAuth: A fine-grained indoor location-based authentication system using wireless networks characteristics. Computers & Security. 2020;89:101683.
31. Rizk H, Torki M, Youssef M. CellinDeep: Robust and accurate cellular-based indoor localization via deep learning. IEEE Sensors Journal; c2018.
32. Tian Y, Denby B, Ahriz I, Roussel P, Dreyfus G. Robust indoor localization and tracking using GSM fingerprints. EURASIP Journal on Wireless Communications and Networking; c2015.
33. Tiglao NM, Alipio M, Cruz RD, Bokhari F, Rauf S, Khan SA. Smartphone-based indoor localization techniques: State-of-the-art and classification. Measurement; c2021.
34. Jawandhiya P. Hardware design for machine learning. International Journal of Artificial Intelligence and Applications (IJAIA); c2018.
35. Aslan MF, Unlersen MF, Sabanci K, Durdu A. CNN-based transfer learning–BiLSTM network: A novel approach for COVID-19 infection detection. Applied Soft Computing. 2021;98:106912.
36. Wu YC, Feng JW. Development and application of artificial neural network. Wireless Personal Communications. 2018;102(2):1645-1656.
37. Cireşan DC, Meier U, Masci J, Gambardella LM, Schmidhuber J. High-performance neural networks for visual object classification. arXiv preprint arXiv:1102.0183; c2011.
38. Goodfellow I, Lee H, Le Q, Saxe A, Ng A. Measuring invariances in deep networks. Advances in Neural Information Processing Systems. 2009;22:646-654.
39. Roth HR, Yao J, Lu L, Stieger J, Burns JE, Summers RM. Detection of sclerotic spine metastases via random aggregation of deep convolutional neural network classifications. In: Recent advances in computational methods and clinical applications for spine imaging. Cham: Springer; c2015. p. 3-12.
40. Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov RR. Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint. arXiv:1207.0580; c2012.
41. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research. 2014;15(1):1929-1958.
42. Hinton GE, Osindero S, Teh YW. A fast learning algorithm for deep belief nets. Neural Computation. 2006;18(7):1527-1554.
43. Salakhutdinov R. Learning deep generative models. Annual Review of Statistics and Its Applications. 2015;2:361-385.