Toward a Cloud based Disease Diagnosis System Using Sequential Quadratic Programming Approach

Ali Hussein Shamman Al-Safi
Computer Techniques Engineering Department, Al-Mustaqbal University College, Hillah, Babil, Iraq.

Zaid Ibrahim Rasool Hani
Computer Techniques Engineering Department, Al-Mustaqbal University College, Hillah, Babil, Iraq.

Ahmed A Hadi
Computer Techniques Engineering Department, Al-Mustaqbal University College, Hillah, Babil, Iraq.

Musaddak M. Abdul Zahra*
Computer Techniques Engineering Department, Al-Mustaqbal University College, Hillah, Babil, Iraq.
Electrical Engineering Department, College of Engineering, University of Babylon, Hilla, Babil, Iraq. E-mail: musaddaqmahir@mustaqbal-college.edu.iq

Wael Jabbar Abed Al-Nidawi
Computer Techniques Engineering Department, Al-Mustaqbal University College, Hillah, Babil, Iraq.

Received May 08, 2021; Accepted September 10, 2021
ISSN: 1735-188X
DOI: 10.14704/WEB/V18I2/WEB18368

Abstract

The Internet of Things (IoT) relates to the process of utilizing computer networks to plan and model Internet-connected things. The Internet of Things (IoT)-based m-healthcare technologies have provided multi-dimensional functionality and real-time resources over the last few years. These apps provide millions of individuals with a forum to get wellness alerts for a healthy lifestyle constantly. Several aspects of these systems have been revitalized with the introduction of IoT devices in the healthcare sector. This work proposed a data-driven disease signal analytics by inventing a novel combination learning approach. The proposed Combination learning integrates different machine learning models to price disease signal for different options by leveraging the availability of a large amount of data through solving a sequential quadratic programming problem. The proposed approach demonstrates its superiority in prediction accuracy and strong model independence by overcoming traditional model-driven

approaches' generalization issue. The findings illustrate the efficacy of the task for an effective disease signal diagnosis. It could be a modern and useful health approach to adopt the proposed procedure with potential changes and incorporate it into a low-cost unit.

Keywords

Cloud Computing, Internet of Things, Healthcare, Diagnosis, Machine learning, Sequential Quadratic Programming Introduction.

Introduction

The Internet of Things (IoT) relates to the process of utilizing computer networks to plan and model Internet-connected things. IoT means that it is easier to have a wide range of less efficient devices, including wrist bands, air-conditioners, umbrellas, and refrigerators, instead of providing a limited number of powerful electronic devices such as computers, tablets, and mobile. Sensor-assisted computer processors intelligently program typical human-usable items such as air fresheners and cars and generate production in the actual environment, both of which are integrated into an ordinary object that can be used (Abdulzahra, 2019; Akhtar et al., 2015; Rasool et al., 2021; Ye et al., 2021; Zahra et al., 2020). Therefore, wired devices or objects can process and interact beyond the specifications of basic devices, including an average lamp and umbrella, and may link buildings by network connectivity. Except for a name and personality, these improved IoT items have the technical thinking power to fulfill the assigned task. Since IoT is run through a broad spectrum of Internet networks, the word "Ubiquitous computing" varies from IoT (Anjos et al., 2013; G. Zhang et al., 2020). Inputs from a person or a living thing may be received from the "Thing" or object existing in the physical world and convert such data to the Internet for processing and data collection. For instance, the thread left behind, the number of stitches sewn, and roughly how many stitches can be registered by a sewing machine, which rendered possible throughout the specified time by utilizing sensors to save the output displayed by the object. In the sensors, "Actuators" may demonstrate the outputs to the human environment by linking the objects in the field. The gathered data triggers some of these outputs and is interacted with by the Internet. A sewing machine might signal that it runs out of threads, and substitution must be performed (Breiman, 2001; Xu et al., 2016).

The mixture of cloud and IoT-based web apps performs better in terms of performance relative to ordinary cloud-based apps. These variations may be used for new uses, including military, medical, and banking apps. The cloud-based IoT method can help supply medical applications with effective resources to track and view information from every remote

venue. The requisite details, such as regular adjustments in health criteria, are obtained by IoT-centric healthcare apps on time. Throughout a regular time, it updates the magnitude of the diagnostic parameters. Besides, IoT sensors and sensor-related diagnostic criteria can be used successfully to detect the illness at the proper period before the serious condition is reached. (Wen et al., 2015).

In the decision-making method, machine learning algorithms play a significant role, even managing a very large amount of data. Implementing data collection approaches for particular fields requires identifying categories of data such as velocity, variety, and length. A neural network model, categorization model, and clustering system are used in the traditional data processing model, and efficient algorithms are implemented. Data with unique data kinds may be created from multiple sources, and methods need to be built to manage data features. The vast quantities of IoT resources produce the needed data in real-time without any issues, such as scalability, speed, and identifying the perfect data model. These are all known to be one of the IoT's big problems. These are all challenges, though, which provide a large number of possibilities for new technologies. A large amount of big data, which has various types of data, such as image, document, and categorical data using IoT devices as input data, has been obtained in this work. Such details would be safely processed in the cloud setting and accessible by the latest healthcare apps created. To continue with the learning method, a new machine-learning algorithm was used to map the data into two classes:' Usual' and 'Disease Affected' (Boulkenafet et al., 2015; Jain et al., 2005; Jain A, n.d.).

Since learning tasks involved in health diagnosis detection using Cloud and IoT can be more complex than other learning tasks (e.g., speech and voice processing), it is desirable to fully use several ML models to exploit the power of machine learning fully. Third, feature selection is ad-hoc. There is almost no rule to determine which feature should enter machine learning on behalf of health diagnosis. As a result, some important variables (e.g., ask-bid) may not participate in learning (Murdas & Zahra, 2016; Patel et al., 2016).

It is vital to develop a fully data-driven approach that takes advantage of massive amounts of data and conquers the existing limitations. It should be more generalized rather than only work for a specific type of option. It should also exploit machine learning by using multiple models and integrating different prediction results, besides solving the ad-hoc feature selection problem (Liaw & Wiener, 2002; Pan et al., 2007; Ruiz-Blondet et al., 2016; Tsolas & Charles, 2015). However, it can be challenging to attain the goal. To our knowledge, there is no previous work on this topic. Besides data collection, ML model evaluations, and variable selection, several key questions have to be answered: what kinds of ML models

should be used in disease diagnosis? How to integrate different ML models to achieve optimized pricing? This study presents a fully data-driven approach by proposing novel integrative learning to answer the questions. The proposed method integrates different ML models to price disease signals by leveraging massive data collected. The theoretical foundation of integrative learning is built upon the framework of shallow, mid-level, and deep learning proposed in this work. Integrative learning builds an integrative learning system from top-performed ML models upon a large amount of option data. It presents mean-square error (MSE) and sequential quadratic programming weighting methods to integrate ML models. The sequential quadratic programming method models disease diagnosis as a nonlinearly constrained optimization problem and solves it via sequential quadratic programming (Gill & Wong, 2012; Koza & Koza, 1992).

The proposed approach demonstrates superiority to traditional methods in generalization and robustness by providing decent subject prediction as the first full data-driven approach converts subject prediction into a model-free problem by avoiding high complexities in modeling. It can also lower overfitting risks for the multiple ML model usage. On the other hand, the proposed integrative learning can work as a general algorithm to integrate different machine learning models. It will inspire more following work in machine learning integrations in analytics. It was interchangeably using the following terms: pricing and prediction, learning machine, and ML models in our context for the convenience of descriptions.

The remaining parts are arranged as follows: first, the approach suggested is presented; then, the tests are described; finally, the consequences and the potential study for reconsideration.

The Framework of the Proposed Method

In particular, in Figure 1, the suggested procedure is seen. The system proposed comprises three key stages: preprocessing of data, extraction of features, and creating classifiers. In the parts below, these measures are described.

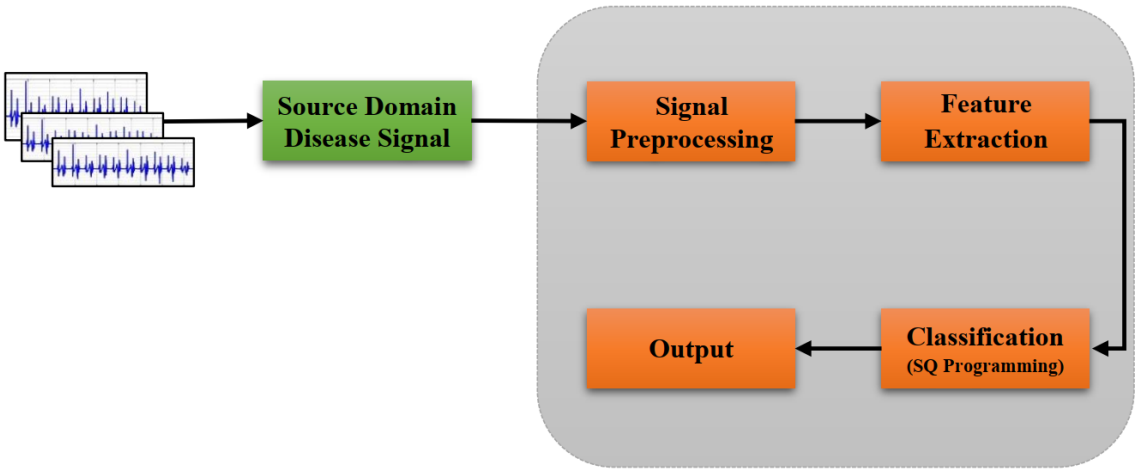

Fig. 1 The architecture of Disease Diagnosis Computer Interface

They are building an integrative learning system for full data-driven subject prediction. It involves three phases: preprocessing and data collection, variable selection, and integrative learning. The data collection and preprocessing stage collects a large scale of option data and cleans the data by removing missing ones or outliers.

The variable selection stage answers the query: 'which variables of an option should enter machine learning?' The integrative learning stage integrates different ML models to get better predictions. It consists of four steps: preparation, probing learning, batch learning, and integration. Preparation normalizes and partitions data into training and test for learning. Probing learning evaluates a set of ML models by using partial training data and identifies top-performing models. Batch learning uses top-performing models to predict subject by using whole training data. Integration aims at integrating batch learning results to get better predictions. It was proposed sequential quadratic programming and meant square error (MSE) weighting for integration, in which the first one models subject prediction as a nonlinear optimization problem (Liaw & Wiener, 2002).

Data Preprocessing

Two procedures were conducted in the preprocessing period. At first, potential noise has been achieved through a clear method in the disease signal. An average filter is used in this process. Besides, because some of the disease signals are not on the same scale, there has been uniformity between the disease signals' sizes. The following method could be used to compute an average filter:

$$\text{Signal}^{\text{Filter}} = \frac{1}{n}\sum_{j=1}^{n} \text{signal}_{j}^{\text{orig}} \qquad (1)$$

Feature Extraction

This phase focuses on statistical measurements and energy coefficients measured for each DWT biorthogonal 2.2 (bior 2.2) decomposition step. In both instances, each channel's characteristics have been calculated and concatenated, thereby achieving a single function vector per instance.

1. Features Depending on DWT

Disease signals are typically non-stationary since they shift rapidly over time, and disease trends provide knowledge regarding particular changes over time why this behavior must be taken into consideration by a representation.

DWT with bior 2.2 as the mother function has been applied for each disease's signal channel as a first option. Applying this transform with a decomposition level j provides a structure with guesstimate vectors CAj and detail coefficients CDj: [CAj,,CDj−1, …,CD1]. It is also potential to identify the optimum decomposition levels' number to be calculated utilizing DWT depending on the size of epochs that required for this dataset was four levels. Table 1 demonstrates the frequencies for each decomposition's level.

The wavelet coefficients, nevertheless, can differ based on the channel for each decomposition level. The Instant Energy Delivery (IWE) and Teager (TWE) were determined to address this issue. It is necessary to provide the same number of characteristics per case after the energy coefficients have been calculated; below are the calculations for these energy distributions.

Table 1 DWT with four decomposition levels, frequency ranges

Level	Frequency
D1	32-64
D2	16-32
D3	8-16
D4	4-8
A4	0-4

- The energy of Instantaneous: this coefficient of energy was a proportion of the disease signal amplitude.

$$f_j = \log_{10}\left(\frac{1}{N_j}\sum_{r=1}^{N_j}\left(\left(w_j(r)\right)\right)^2\right) \quad (2)$$

- The energy of Teager: It represents changes in both the magnitude and frequency of the signal of the disease and is a strong disease identification factor.

$$f_j = \log_{10}\left(\frac{1}{N_j}\sum_{r=1}^{N_j-1}\left(\left(w_j(r)\right)\right)^2 - w_j(r-1) * w_j(r+1)\right) \qquad (3)$$

Instead of providing a vector of magnitude for each stage of decomposition, for each decomposition level, a single value for each is obtained, and the procedure is replicated to every channel. The coefficients of energy were then concatenated from each channel to provide a vector of characteristics to reflect the disease signal.

2. Features Depending on the Statistical Magnitude

Another approach to derive the function from the disease signal has been inspired by statistical characteristics. The signal was defined in such work by calculating a range of 15 statistical magnitudes (9 magnitude and six variations thereof). In the nine works presented, only statistical magnitude (STV) has been regarded without any combinations between them, thereby achieving a set of 9 magnitudes per signal channel of the disease. The magnitude was concatenated after measuring statistical characteristics per channel to provide a single function vector per case.

- STV: mean, maximum, minimum, standard deviation, variance, kurtosis, skewness, sum, median.

Combination Learning

Given input option data $\{xi, yi\}i$ N=1 in which xi is a disease signal sample and its yi is its label, and a set of ML models $\emptyset 1$, $\emptyset 2\ldots$ $\emptyset T$ integrative learning predicts disease signal sample by implementing four steps: preparation, probing learning, batch learning, and integration. The preparation normalizes input option data and further partitions the normalized data into training and test data, in which training data has known disease signal values.

Probing learning tests the 'fitness' between data and the ML models. It identifies top-performing ML models using partial training data for training and another partial training data for the test. Batch learning employs the top-performing ML models to predict subjects by using whole training data. Integration assigns weights to the top-performing ML models' predictions to achieve optimized subject prediction.

1. Performance Measures

We introduce the following performance measures: MSE, max, min, and mean error to evaluate machine learning performance in subject prediction.

Mean square error (MSE) answers the query: what is the average performance of this model?

Given a test dataset $\{x'j, y'j\}$ N=1 in which $y'j$ is the subject of option $x'j$; and a given ML model θ, the MSE is defined as,

$$mse = \frac{1}{n}\sum_{j=1}^{n} \left|y'_j - \hat{g}(x'_j)\right|^2 \quad (4)$$

where $\hat{g}(x'j)$ is the predicted subject: $y'j = \hat{g}(x'j)$; in which $\hat{g}(x)$ is the implicit prediction function for the ML model θ obtained from learning. It is noted that mean error $errmax = \frac{1}{n}\sum n\ j{=}1\ |y'j - \hat{g}(x'j)|$ can also be used as a substitute for MSE in evaluation.

Similarly, the maximum and minimum errors are defined as$errmax$ = max$\{y'j - \hat{g}(x'j)\}i$ n=1, $errmin$ = min$\{y'j - \hat{g}(x'j)\}i$ n=1, respectively. They answer the corresponding queries: what is the worst/best case for this model?

Model performance evaluation. To evaluate two ML models' performance, It checked their average performance first. If they have the same level average performance, check their worst-case performance; if they have the same average and worst-case performance, check their best-case performance. Such a mini-decision tree guarantees a technical evaluation of any two learning models though MSE is usually sufficient.

2. Probing Learning

It is a ’fitness test’ between data and ML models to select the well-performing models. It uses one part of training data for training and the remaining for the test. Since disease signals are known for all training data, they can evaluate each ML model's performance.

It first partitions training data $\{xi, yi\}$ N=1 into probing-training Xp and probing-test data Xp' with a partition threshold η in (0:6; 1); where η means probing-training dataset Xp has 100× η% entries of the whole training data. It is recommended to use a high partition threshold (e.g., η = 0:8) so that the probing-training data can simulate the whole training data better in learning. Then probing learning trains input ML models $\emptyset 1, \emptyset 2 \ldots \emptyset T$ on Xp and calculates their MSEs on $X\ 'p$: Finally, it uses an MSE threshold τ to pick top-performing ones with MSEs< τ.

3. Batch Learning

Batch learning employs the top-performing models selected from probing learning to predict the subject using the whole training data. That is, each selected ML model $\emptyset j$;

j = 1; 2 … l; is trained with the whole training dataset X = {xi, yi}i N=1 Mathematically, it means there are subject prediction functions $\hat{g}j(x)$ for an option sample xk' with the unknown subject, i.e., $\hat{y}'jk=\hat{g}(x'k)$; j = 1; 2 … l; k = 1; 2 … t: Totally, batch learning will generate l predicted volatilities: $\hat{g}1(x'k)$, $\hat{g}2(x'k)$, … $\hat{g}l(x'k)$.

4. Combination

Combination assigns the final subject for an option xk' in test data via the following weighting process. Given subject predictions $\hat{g}1(x'j)$, $\hat{g}2(x'j)$, … $\hat{g}l(x'j)$; assigning the final subject $\hat{g}c(x'j)$ is equivalent to finding the weights $w1$; $w2$ ...wl; such that $wj \geq 0$; $\sum l$ j=1 wi = 1.

$$\hat{g}_c(x'_j) = \sum_{j=1}^{l} w_j \hat{g}_j(x'_j) \qquad (5)$$

This study presents a mean square error (MSE) and sequential quadratic programming weightings. The former assigns weights according to the MSE of each ML model in probing learning. The latter assigns weights by solving a sequential quadratic programming problem.

5. MSE Weighting

The principle of MSE weighting is to assign more weights to those models with smaller MSEs. Using the MSEs from probing learning instead of batch learning is that MSEs cannot be calculated in batch learning because the test data is unknown. It has been present naive weighting approaches as follows.

Naive weighting assigns more weights to the prediction of ML model $\emptyset j$ directly,

$$wj = 1 - \frac{mse_j}{\sum_{j=1}^{l} mse_i} \qquad (6)$$

6. Sequential Quadratic Programming Weighting

The MSE weighting is a static weighting scheme, which assigns the same weights to all options samples. Sequential quadratic programming weighting assigns different weights to options in a dynamic way. Its idea can be sketched as follows. Given an option xj' in test data and l ML model $\emptyset j$; j = 1; 2 … l; in batch learning, It has been firstly finding at least l nearest neighbors of xj' in the training data, say, $x1$; $x2$;…xm; $m \geq l$; each of which has its known subjects $y1$; $y2$;…ym: Then, solve the following problem to look for the weights {wi}i l=1.

$$\sum_{i=1}^{m} \sum_{j=1}^{l} w_j \hat{g}_j(x_j') = y_i, \sum_{j=1}^{l} w_j = 1, w_j \geq 0 \qquad (7)$$

in which $\hat{gj}(xi)$ is the predicted subject of xi from the ML models $\emptyset_j$; j = 1; 2 … l; i =1; 2 … m.

Let $(\hat{g})$ = $\hat{gj}(x'i)$; then the following matrix A ∈ $\Re^{m\times 1}$ stores all predictions of the neighbors$x1$; $x2$;…xm.

$$A = \begin{bmatrix} \hat{g}_1(x_1) & \hat{g}_2(x_1) & \cdots & \hat{g}_l(x_1) \\ \hat{g}_1(x_2) & \hat{g}_2(x_2) & \cdots & \hat{g}_l(x_2) \\ \cdots & \cdots & \cdots & \cdots \\ \hat{g}_1(x_m) & \hat{g}_2(x_m) & \cdots & \hat{g}_l(x_m) \end{bmatrix} \qquad (8)$$

The need to solve the problem:

$$A \begin{bmatrix} \hat{g}_1(x_1) & \hat{g}_2(x_1) & \cdots & \hat{g}_l(x_1) \\ \hat{g}_1(x_2) & \hat{g}_2(x_2) & \cdots & \hat{g}_l(x_2) \\ \cdots & \cdots & \cdots & \cdots \\ \hat{g}_1(x_m) & \hat{g}_2(x_m) & \cdots & \hat{g}_l(x_m) \end{bmatrix} \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_l \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} \qquad (9)$$

by satisfying the constraints: $wj \geq 0$; $\sum_{j=1}^{1} wi = 1$.

Which is equivalent to solving the nonlinear optimization problem:

$$min\left|\sum_{i=1}^{m} \sum_{j=1}^{l} WJ - y_j \hat{g}_{ij}\right|^2 \qquad (10)$$

Let = [$w1$; $w2$ … wl]; and y = [$y1$; $y2$ … yl]; It has been rewriting it as a standard sequential quadratic programming format:

$$\begin{aligned} &\min_{w} f(w), st \\ &h(w) = 0 \\ &g(w) \geq 0 \end{aligned} \qquad (11)$$

where $f(w) = \| Aw - y \|_2^2$; $h(w) = (w \cdot \vec{1})$; $g(w) = w$ is a vector in $\Re^l$ in which all entries are 1.

$$\begin{aligned} &\min_{d} \nabla f(w_k)^T d + \frac{1}{2} d^T B_k d, s.t \\ &\quad h(w_k)\nabla h(w_k) d = 0 \\ &\quad g(w_k)\nabla g(w_k) d \geq 0 \end{aligned} \qquad (12)$$

where Bk is the Hessian matrix of $f(w)$ at the kth iteration. SP solves a QP problem in each iteration to get the direction d; and calculates the next iteration point: $wk+1 = wk + d$ until d approximates a zero vector.

Results

By testing with two sets of various functions, this segment intends to demonstrate the effects. The dataset utilized has been detailed, and by utilizing multiple functions, the findings are retrieved later. In comparison, the findings gained when a decreased range of channels are used as demonstrated. For assessing the proposed spatiotemporal disease signal's effectiveness, It has been performed a set of experiments on ECG/EEG signals obtained from the Physio net ECG/EEG database. It has been adopting two such databases and presents the characteristics of ECG/EEG signals obtained from the said database (Z. Zhang et al., 2012). It should be noted that the k-fold method to divide the data into our training and testing phase. In this method, the best number of a fork is considered 10.

Machine Learning Models in Subject Prediction

It categorizes machine learning into shallow learning, mid-level learning, and deep learning in this study.

Shallow, Mid-level, and Deep Learning

Shallow learning does not have a serious learning topology, where each input data point is viewed as a learning unit. It has a simple learning mechanism. For example, k-NN is a typical shallow learning ML model with no rigorously mathematical model support for its instance learning mechanism.

A complicated learning topology and mechanism characterize deep learning. A deep learning machine has its learning unit: a neuron that simulates a human brain neuron to handle information processing. Each neuron has its own activation functions (e.g., rectified linear unit function ('relu')) in information processing. The neurons can demonstrate very different and complicate topologies in connections, which may change dynamically. MLP is a typical deep learning model with complicated learning topologies and serious learning mechanisms (e.g., stochastic gradient descent learning) (Gill & Wong, 2012; Kingma & Ba, 2014).

An ML model from mid-level learning has the same level of complicated learning mechanism as a deep-learning machine. However, they have a rigorous learning theory

(e.g., kernel-based learning) (Naganna et al., 2019; Oleiwi et al., 2018; Shawe-Taylor & Cristianini, 2000). Also, its learning topology may not be as complicated as that of a deep learning machine. For example, support vector machine, and (SVM), gradient boosting (GB), random forests (RF), or other ensemble learning methods all belong to mid-level learning (Breiman, 2001; Friedman, n.d.).

Probing Learning Results

It has been selected 9 ML models ranging from shallow learning to deep learning in probing learning. They include k-NN, decision trees (DT), Support vector machine (SVM) with linear and RBF kernels, Bayesian Ridge (Bayesian), Adaptive Boosting (AdaBoost), Bagging with decision tree as an estimator, and Multi-layer perceptron (MLP) (neural network) (Breiman, 1996; Louppe & Geurts, 2012). It is found that multi-hidden-layer MLP does not work well as a single hidden layer one. It is probably because the input data scale is not large enough to employ a deep neural network (DNN). On the other hand, a neural network with one hidden layer is enough to approximate the subject prediction function (Mokhtari & Ribeiro, 2013). It has been employed MLPs with different 200, 300, and 500 hidden nodes, respectively. Each of them uses a 'relu' activation function under quasi-Newton solver: 'LBFGS' with maximum iterations: 106 and 104 tolerance (Hornik, 1991). It has been employed as the quasi-Newton solver instead of other gradient learning (e.g., Adam) because of its second-order convergence speed.

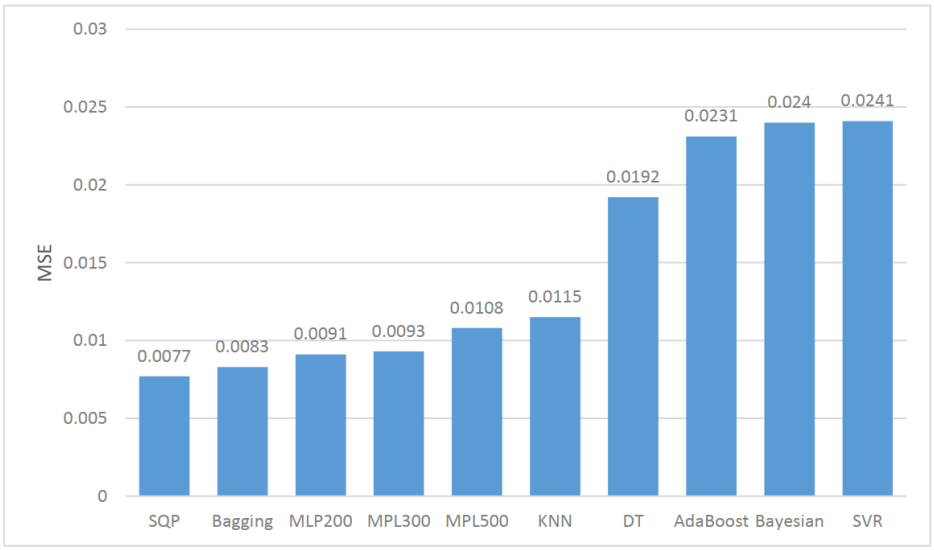

Figure 2 The distributions of MSE of 12 ML models in probing learning, where SVR linear and SVR represent SVM regression with linear and 'rbf ' kernels, respectively

Figure 2 shows MSE values for the different ML models in probing learning, indicating sequential quadratic programming, bagging, and MLPs are leading performers. The ML

model parameter settings follow defaults generally to avoid over tuning. The sequential quadratic programming model has a learning rate of 0:005; the boosting phases' number to carry out is set as 1000, the optimum depth of each decision tree is set at 8, and the lowest required specimens' number to be at a leaf node is 2. The bagging model has ten decision tree estimators and creates its training datasets via bootstrap. The MLPs lower their performance in probing learning with the increase of hidden neurons. It suggests that more hidden units may even lower neural network's predictability. As a shallow learning method, k-NN chooses Euclidean distance, picks three nearest neighbors, and uses inverse distance weights in regression. It outperforms some mid-level learning models such as linear-SVM, RBF - SVM, AdaBoost, decision trees (DT), and Bayesian ridge (Bayesian).

Table 2 illustrates all models' performance in probing learning on behalf of MSE, max, min, and mean errors. It indicates that MSE or mean error can evaluate each model's performance better than max or min error, although the max (min) error can indicate how bad (well) a model can diverge (converge) in subject prediction. The max errors stay at the same level for almost all models except for MLP (500), showing an obvious divergence, but the min errors demonstrate more 'ad-hoc' values for different models.

MSE threshold selection employed a genetic rule to select the MSE threshold τ in probing learning. It is set as the mean of all MSEs in probing learning adding the half of the standard deviation of MSE, i.e., $\tau = E(MSE) + \frac{\text{std}(MSE)}{2}$: That is τ = 0:0179 under this situation, which means the ML models with MSEs larger than this cutoff will be dropped in batch learning. A total of 8 ML models entered batch learning, with the following four models dropped: decision trees (DT), AdaBoost, Bayesian ridge regression (Bayesian), and linear-SVM.

1. Batch Learning

Batch learning uses the selected top-performing models to predict the test data subject using the whole training data. Thus, there are eight predicted subject: $\hat{g}1(x)$; $\hat{g}2(x)$ …$\hat{g}8(x1)$; for a given option x in the test data. It has not been known as the MSE of each model in batch learning because the test data subjects are supposed to be unknown. It has been only known they should work better than probing learning because more training data is used for the top players in probing learning.

To answer the query, what happens in batch learning? Figure 3 compares the selected models' performance in batch and probe learning in terms of MSEs, max (min) errors, and mean errors. It indicates that almost all selected ML models in batch learning improve their predictions except the DT according to the measures. The model performance in batch

learning follows its ranking orders in probing learning globally. It validates the 'correctness' of probing learning estimation. Interestingly, the three neural network models demonstrate an 'inverse' performance in batch learning compared to probing learning, where the models with more hidden nodes demonstrate slight advantages in performance.

Furthermore, bagging, sequential quadratic programming, and MLP-500 (a neural network with 500 hidden neurons) achieve much lower max error in batch learning. Such results suggest that more training data improves the average performance and contributes to their worst-case scenarios. Also, batch learning's minimum errors in batch learning show an equivalent or better performance in batch learning than probe learning.

Table 2 ML model performance in probing learning

Methods	MSE	Max Error	mirror
SQ Programming	0.0077	0.5893	0.0589
Bagging	0.0083	0.6295	0.0000
MPL200	0.0091	0.5753	0.0000
MLP300	0.0093	0.5781	0.0000
MLP500	0.0100	1.3170	0.0000
KNN	0.0115	0.5882	0.0000
DT	0.0192	0.6712	0.0000
AdaBoost	0.0232	0.5711	0.0002
Bayesian	0.0240	0.6620	0.0006

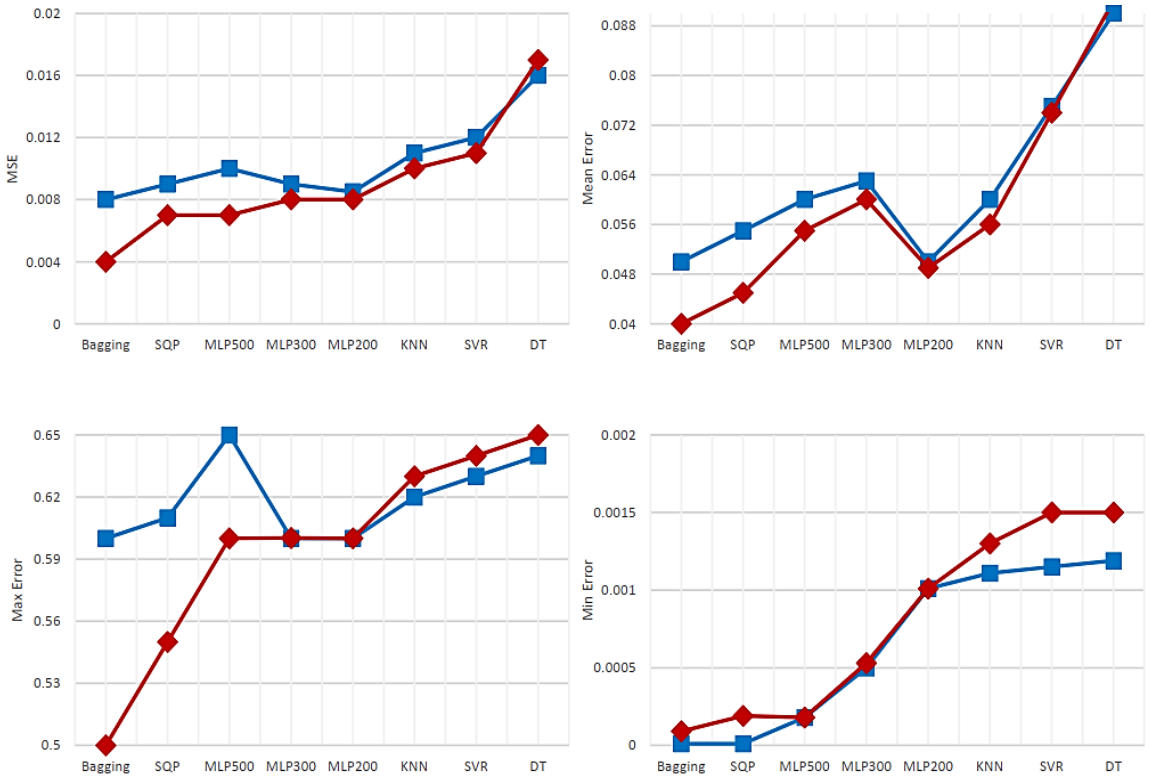

Figure 3 The performance comparisons of the selected models in batch and probe learning in terms of MSEs, mean errors, and max (min) errors

2. Sequential Quadratic Programming: Final Subject Prediction

In addition to MSE and sequential quadratic programming weighting methods, 'mean' and 'median' weighting to calculate final subjects using the mean and median of the predicted subjects. Mathematically, they are equivalent to inferring the following prediction functions: $\widehat{GC}_{(x)} = \frac{1}{l}\sum_{j=1}^{l} \hat{g}_j(x), \widehat{GC}_{(x)} = \text{median}\{\hat{g}_j(x)\}_{j=1}^{l}$ correspondingly. The former assigns the same weights to all subject predictions, and the latter assigns weights such that the final subject to be the median of all predictions in batch learning.

Table 3 The MSE comparisons of MSE weighting, sequential quadratic programming weighting, and the three top-performing models in batching learning. The sequential quadratic programming weighting in integrative learning achieves the best performance

Methods	No. channels	MSE	No. channels	MSE
SQ	2	0.012	10	0.0066
	4	0.0099	12	0.0062
Programming	6	0.0075	14	0.0059
	8	0.0069	16	0.0055
Bagging	2	0.12	10	0.013
	4	0.0196	12	0.0099
	6	0.0185	14	0.0085
	8	0.016	16	0.0077
MLP500	2	0.1602	10	0.0162
	4	0.142	12	0.0123
	6	0.1126	14	0.0096
	8	0.1025	16	0.0081
Naive	2	0.0126	10	0.0096
	4	0.012	12	0.0075
	6	0.0109	14	0.0065
	8	0.01	16	0.0062

Table 3 illustrates the three top-performed ML models' performance in batch learning, MSE weighting, and sequential quadratic programming weighting on behalf of MSEs. It shows that the proposed Combination learning achieves better performance than Bagging and MLP with 500 neurons in the hidden layer. The sequential quadratic programming weighting outperforms other integration methods by achieving approximately 25% improvements in the average performance of three single ML models. All integrative learning methods achieve better predictions than the single ML model predictions.

Conclusion

In this study, It has been proposed a fully data-driven approach using integrative learning to predict disease signals. The integrative learning approach takes advantage of the large amounts of option data and exploits multiple ML models' learning potential. It overcomes

the weakness of the traditional model-driven approach by enhancing the generalization of disease signal pricing.

References

Abdulzahra, M.M. (2019). Novel Anti-Collision Algorithm in RFID Tag Identification Process. *Computer Science On-Line Conference*, 152–169.

Akhtar, Z., Micheloni, C., & Foresti, G.L. (2015). Biometric liveness detection: Challenges and research opportunities. *IEEE Security & Privacy*, *13*(5), 63–72.

Anjos, A., Chakka, M.M., & Marcel, S. (2013). Motion-based counter-measures to photo attacks in face recognition. *IET Biometrics*, *3*(3), 147–158.

Boulkenafet, Z., Komulainen, J., & Hadid, A. (2015). Face anti-spoofing based on color texture analysis. *IEEE International Conference on Image Processing (ICIP)*, 2636–2640.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, *24*(2), 123–140.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32.

Friedman, J.H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, *29*(5),1189-1232.

Gill, P.E., & Wong, E. (2012). Sequential quadratic programming methods. In *Mixed integer nonlinear programming*, 147–224.

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural Networks*, *4*(2), 251–257.

Jain, A.K., Ross, A., & Uludag, U. (2005). Biometric template security: Challenges and solutions. *13th European Signal Processing Conference*, 1–4.

Jain, A.K., Ross, A., & Prabhakar, S. (2004). An introduction to biometric recognition. *IEEE Transactions on circuits and systems for video technology*, *14*(1), 4-20.

Kingma, D.P., & Ba, J. (2014). Adam: A method for stochastic optimization. *ArXiv Preprint ArXiv:1412.6980*.

Koza, J.R., & Koza, J.R. (1992). *Genetic programming: on the programming of computers by means of natural selection* (Vol. 1). MIT press.

Liaw, A., & Wiener, M. (2002). Classification and regression by randomForest. *R news*, *2*(3), 18-22.

Louppe, G., & Geurts, P. (2012). Ensembles on random patches. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 346–361.

Mokhtari, A., & Ribeiro, A. (2013). Regularized stochastic bfgs algorithm. *IEEE Global Conference on Signal and Information Processing*, 1109–1112.

Murdas, I.A., & Zahra, M.M.A. (2016). Simulation of Ultra High Bit Rate DWDM/PDM/SDM System Using FMF Supporting 20 Spatial and Polarization Modes Enabling Multi-Mode Splicers with QPSK Modulation Format. *International Journal of Engineering and Technology*, *6*(12).

Naganna, S., Deka, P., Ghorbani, M., Biazar, S., Al-Ansari, N., & Yaseen, Z. (2019). Dew Point Temperature Estimation: Application of Artificial Intelligence Model Integrated with Nature-Inspired Optimization Algorithms. *Water*, *11*(4). https://doi.org/10.3390/w11040742

Oleiwi, S., Jalal, S., Hamed, S., Ozgur, S., Zaher, K., & Yaseen, M. (2018). Precipitation pattern modeling using cross-station perception: regional investigation. *Environmental Earth Sciences*, *77*(19), 1-11. https://doi.org/10.1007/s12665-018-7898-0

Pan, G., Sun, L., Wu, Z., & Lao, S. (2007). Eyeblink-based anti-spoofing in face recognition from a generic webcamera. *IEEE 11th International Conference on Computer Vision*, 1–8.

Patel, K., Han, H., & Jain, A.K. (2016). Cross-database face antispoofing with robust feature representation. *Chinese Conference on Biometric Recognition*, 611–619.

Rasool, Z.I., Abd Ali, R.S., & Abdulzahra, M.M. (2021). Network Management in Software-Defined Network: A Survey. *IOP Conference Series: Materials Science and Engineering*, *1094*(1).

Ruiz-Blondet, M.V, Jin, Z., & Laszlo, S. (2016). CEREBRE: A novel method for very high accuracy event-related potential biometric identification. *IEEE Transactions on Information Forensics and Security*, *11*(7), 1618–1629.

Shawe-Taylor, J., & Cristianini, N. (2000). *Support vector machines*. Cambridge university press Cambridge, 2.

Tsolas, I.E., & Charles, V. (2015). Incorporating risk into bank efficiency: A satisficing DEA approach to assess the Greek banking crisis. *Expert Systems with Applications*, *42*(7), 3491–3500.

Wen, D., Han, H., & Jain, A.K. (2015). Face spoof detection with image distortion analysis. *IEEE Transactions on Information Forensics and Security*, *10*(4), 746–761.

Xu, Y., Price, T., Frahm, J.M., & Monrose, F. (2016). Virtual u: Defeating face liveness detection by building virtual models from your public photos. *25th {USENIX} Security Symposium ({USENIX} Security 16)*, 497–512.

Ye, L., Jabbar, S.F., Zahra, M.M.A., & Tan, M.L. (2021). *Bayesian Regularized Neural Network Model Development for Predicting Daily Rainfall from Sea Level Pressure Data : Investigation on Solving Complex Hydrology Problem*.

Zahra, M.M. A., Mohsin, M J., & Abdul-Rahaim, L.A. (2020). Artificial intelligent smart home automation with secured camera management-based GSM, cloud computing, and arduino. *Periodicals of Engineering and Natural Sciences*, *8*(4), 2160–2168.

Zhang, G., Ali, Z.H., Aldlemy, M.S., Mussa, M.H., Salih, S. Q., Hameed, M.M., Al-Khafaji, Z. S., & Yaseen, Z.M. (2020). Reinforced concrete deep beam shear strength capacity modelling using an integrative bio-inspired algorithm with an artificial intelligence model. *Engineering with Computers*, 1–14.

Zhang, Z., Yan, J., Liu, S., Lei, Z., Yi, D., & Li, S.Z. (2012). A face antispoofing database with diverse attacks. *5th IAPR International Conference on Biometrics (ICB)*, 26–31.

James, L. (2016). A psychobiological model for managing student engagement in online courses using gamification principles. *Webology*, *13*(1), 1-23.