A Service-based Joint Model Used for Distributed Learning: Application for Smart Agriculture

Dixon Vimalajeewa, Chamil Kulatunga, Donagh P. Berry, Sasitharan Balasubramaniam

Abstract—Distributed analytics facilitate to make the data-driven services smarter for a wider range of applications in many domains, including agriculture. The key to producing services at such level is timely analysis for deriving insights from reliable data. Centralized data analytic services are becoming infeasible due to limitations in the Information and Communication Technologies (ICT) infrastructure, timeliness of the information, and data ownership. Distributed Machine Learning (DML) platforms facilitate efficient data analysis and overcome such limitations effectively. Federated Learning (FL) is a DML methodology, which enables optimizing resource consumption while performing privacy-preserved timely analytics. In order to create such services through FL, there needs to be innovative machine learning (ML) models as data complexity as well as application requirements limit the applicability of existing ML models. Even though NN-based models are highly advantageous, use of NN in FL settings is limited with thin clients (with less computational capabilities) and high-dimensional data (with large number of model parameters). Therefore, in this paper, we propose a novel Neural Network (NN)- and Partial Least Square (PLS) regression- based joint FL model (FL-NNPLS). Its predictive performance is evaluated under sequential- and parallel-updating based FL algorithms in a smart farming context for milk quality analysis. Smart farming is a fast-growing industrial sector which requires effective analytics platforms to enable sustainable farming practices. However, the use of advanced ML techniques are still at a early stage for improving the effectiveness of farming practices. Our FL-NNPLS approach performs and compares well with a centralized approach and demonstrates state-of-the-art performance.

Index Terms—Decentralized Machine Learning, Federated Optimization, Neural Network, Data Imbalance, MIRS Milk Quality Predictions.

1 Introduction

Rapid adoption of connected technologies such as Internet of Things (IoT) and Cloud Computing are propelling the advancement of data-driven services in many sectors such as for sustainable intensification of food production in agriculture and for personalised/controlled delivery of drugs in health-care. To deliver such services, data analytic frameworks are essentially required to be integrated to the information and communication technologies (ICT) based infrastructures to extract insights and then to communicate effectively with the end-users. The reason behind this is such integrated systems can utilise heterogeneous and massive datasets while enabling granular analytics with continuous changes to produce timely and accurate insights. Hence, an advanced analytical platform coupled with efficient Machine Learning (ML) techniques is required to improve reliability and timeliness of such services [1], [2], [3]. Therefore, this study focuses on proposing an effective analytical framework based on Distributed ML (DML) models coupled with IoT and Cloud Computing infrastructures.

Most sensor technologies and IoT platforms provide services to collate and store vast quantities of data produced from geographically distributed sources. As most computational facilities for analyzing such data reside in centralized data centers (e.g., cloud), where data will be consolidated as single large datasets. Then analytics can subsequently be performed at the central location, which is referred to as Centralized ML (CML). Numerous studies have shown that CML is highly advantageous for developing new hypothesis, as it enables improved learning accuracy [4] and model acceptability [5]. However, data centralization is feasible only when the communication and computational capabilities of the data centers are not limited. At the same time, if data owners are not reluctant to sharing data with CML systems due to data privacy, security, and ownership concerns [6]. These limitations hinder real-time decision making, which is crucial in providing timely services. A promising approach to solve this is Distributed ML (DML), as it facilitates the development of more advanced intelligent systems by incorporating various systems, technologies, and ML techniques in near-realtime. Therefore, to overcome the limitations in CML, there is a growing need for effective DML approaches equipped with functionalities such as data-protection and optimum use of available resources.

DML synchronously or asynchronously executes data-

- *D. Vimalajeewa is with the Telecommunications Software and Systems Group, Waterford Institute of Technology, Waterford, Ireland, E-mail: dvimalajeewa@tssg.org.*
- *C. Kulatunga is with the University College Dublin, Ireland, E-mail : chamilkul@gmail.com.*
- *D. P. Berry is with the Teagasc, Animal & Grassland Research and Innovation Centre, Moorepark, Fermoy, Co. Cork, Ireland, E-mail : Donagh.Berry@teagasc.ie.*
- *S. Balasubramaniam is with Telecommunications Software and Systems Group, Waterford Institute of Technology, Waterford, Ireland, E:mail sasib@tssg.org.*

This work was supported by the Science Foundation Ireland (SFI) projects PrecisionDairy (ID: 13/1A/1977) and VistaMilk (ID: 16/RC/3835), and the Horizon 2020 GenTORE project.

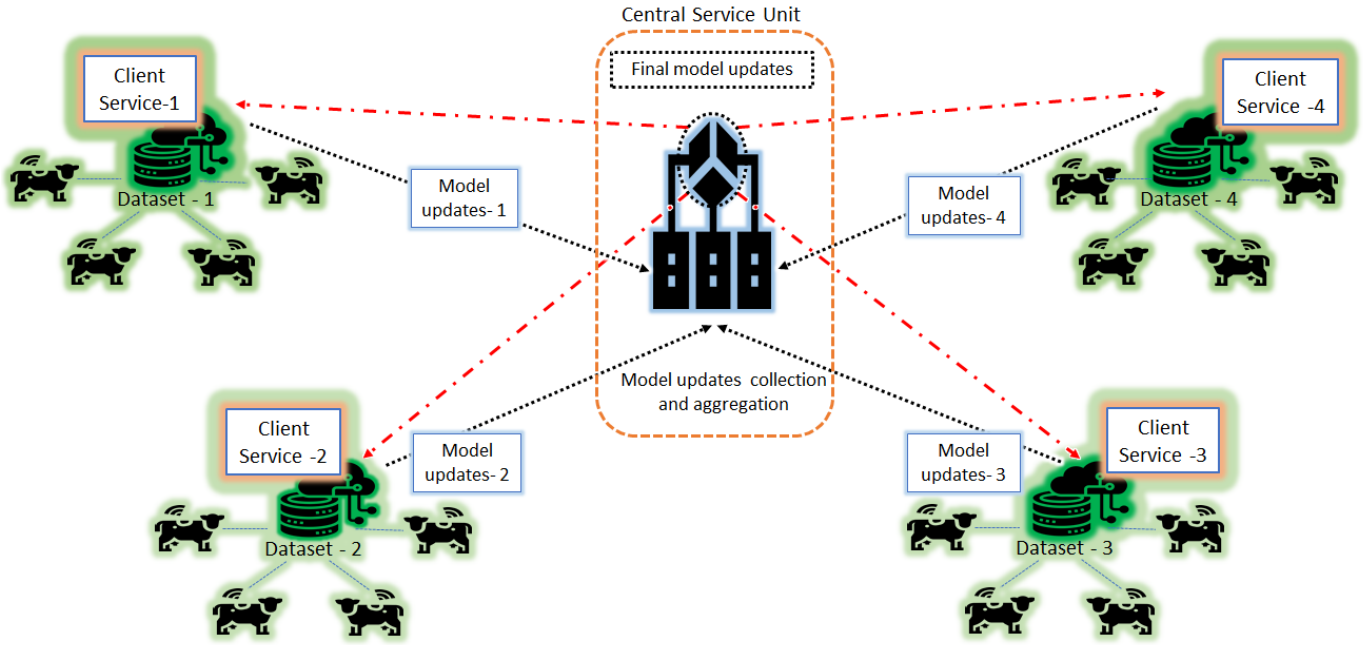


Fig. 1. A FL architecture based on smart dairy farming: (1) Every client (i.e., farm) collects data (e.g., MIRS of milk samples) and trains a ML model and sends the updated model to the central service unit, (2) The central service unit which derives the final model updates by aggregating the model updates of clients, (3) Clients download the final model updates and then update their local ML model to perform inferences.

intensive analytical applications across geographically distributed processing units. Therefore, it captures real-time dynamics in order to make timely decisions for supporting services effectively [7]. For instance, Fog computing paradigm is emerging as a technological enabler for DML, since the analytical process is discovered and offloaded to a node that is in close proximity to the data source [8]. This has led to a growing interest to focus on developing new distributed optimization techniques [9]. Such techniques are Split Learning [10] and Federated Learning (FL) [11]. Among them FL enables optimization algorithms to be federated in order to make real-time decisions without moving data away from the source. This also minimises data privacy concerns and resource consumption (this will be elaborated in a subsequent section).

The key to practicing such DML techniques is having effective ML models which can be trained dynamically (sometimes using thin clients with less computational capabilities like mobiles and tablets) for capturing underlying updated data. The data collected by modern day sensor technologies and IoT platforms is usually highly dynamic, complex, large in size, and highly dimensional which can lead to noise accumulation, multi-correlation, and heavy computational costs. Consequently, such characteristics limit the applicability of commonly used simple ML models such as Least Square Regression (LSQR), Principal Component Analysis (PCA), Partial Least Square Regression (PLSR), and Neural Network (NN). For example, LSQR, PLSR, and NN techniques have extensively been used in various data analytical applications, but fails under certain conditions, thus requiring further enhancements. That is, although PLSR overcomes the limitations of multi-colinearity and high-dimensionality in LSQR fitting, the predictive accuracy is limited due to its inability in capturing complex functional relationships, such as non-linearity [12]. A promising alternative to mitigate such limitations is NN models, as they are becoming increasingly feasible and also actively being used for a wide range of applications. However, selecting optimal NN configurations by using techniques such as grid search and random search depends on the problem, resource availability as well as the experience of the users. With high dimensional data like MIRS milk spectra, minimising model parameters while training a NN model in a thin client is also challenging. It may be inefficient in terms of time and resource consumption as most IoT sensor-based DML applications are resource constrained and timely delivery of the outcomes is vital. This means that the NN approach may not be an efficient alternative in all the cases.

Consequently, modern studies are exploring innovative ML models which are, for instance, scalable, adaptive, lightweight, and computationally inexpensive. It has already been proven that the combined use of NN models with conventional ML models such as PLSR and PCA (known as hybrid models) has the potential to alleviate the barriers that arise when they are used separately, while providing accurate outcomes. However, this scenario has been used in a limited number of studies such as [12], [13], and also under CML settings. To date, there has been no study that has investigated its use within DML using high-dimensional data, and in particular under FL. Therefore, in this study, we propose a joint novel model of coupling NN with PLSR models that can be used for FL with dimensionality reduction. It enables overcoming a number of issues when they are used separately along with limitations in DML such as data sharing issues. We demonstrated the effectiveness of FL in the context of smart dairy farming by using Mid-Infrared Spectroscopy (MIRS) analysis of milk samples, which are routinely used for the quantification of milk quality (see Figure 1). Milk quality analysis particularly plays a vital role in the dairy industry [14], where micro-nutrients of milk

components monitored by robotic milking machines can be analyzed using ML techniques to determine several important insights such as health issues [15], nutrient deficiencies [16], and social behaviour [17]. These insights provide support in terms of intensifying sustainable farming practices such as controlled delivery of nutrients and fertilizers and reducing cost while increasing the profit. However, MIRS-based chemometric analysis have extensively used PCA, LSQR, and PLSR models in earlier studies such as [18], [19]. Joint ML models have not generally been used under FL settings for any analytical purposes, and particularly in the field of agriculture. Therefore, our main contributions of the study is summarized as follows:

- By considering the limitations of the LSQR, PLSR, and NN techniques, introduce a joint ML model known as NNPLS, which combines NN and PLSR techniques, overcoming those limitations.
- Evaluate two federation approaches, which are sequential and parallel, where they can be used to federate the NNPLS model, and apply these approaches to a Smart Dairy Farming application. Specifically, the application is for Mid-Infrared Spectroscopy (MIRS) milk quality data analysis.
- Evaluate and compare the predictive performances of the NNPLS model under CML and DML (FL) settings.
- Ensure that the NNPLS model based FL has comparable performance to the state-of-the-art methods by comparing FL performance of the NNPLS model to an advanced ML technique, which is Convolutional NN (CNN) model.
- Discuss the advantages and challenges of the NNPLS-based FL while proposing an alternative to overcome the imbalance issues in FL, including few potential future directions that this study may extend further.

The remainder of the paper is organnized as follows. Section 2 summarizes the state-of-the-art in distributed machine learning, including the discussion of the FL process followed by the NNPLS model. FL approaches are discussed in detail in Section 3. Section 4 presents the performances of NNPLS under FL settings while Section 5 and 6 discusses the benefits and limitations of the FL-NNPLS approach and concludes the paper, respectively.

2 State-of-the-art on Federated Learning

The progressive advancements of distributed nature of IoT and cloud technologies have contributed to development of smarter services that are found in various sectors such as transportation [20], healthcare [21], agriculture [22], [23], and energy [24]. Therefore, this section provides an overview of the relevant literature focusing on the attempts made in advancing DML mechanisms, including the importance of FL in DML, and the involvement of deep learning and hybrid ML models in ML.

2.1 Distributed Machine Learning (DML)

CML has traditionally been the dominant ML approach. However, with the growing prevalence of big data, CML approaches are facing increasing challenges in collecting and processing massive datasets due to constraints in resources in modern ICT platforms [6], and in particular low powered devices. Consequently, the concept of DML is gaining traction for a wide range of applications such as image classification [9], smart healthcare [21], and smart agriculture [22]. This has led to continued development in new DML algorithms with high accuracy and fast convergence rates [9]. As a result, most of the CML algorithms were converted into DML models, e.g., Bayesian networks, decision trees, and support vector machine [25]. Nevertheless, to support the ever-increasing complexity of data and their features, research in ML has focused on incorporating learning functionalities, and one example is Deep Learning [6].

In employing learning functionalities in a broader spectrum of DML applications, the MapReduce computational model proposed in [26] migrated computations towards the source of the data, which significantly reduced the communication requirements of big datasets even when Hadoop stored data in commodity hardware clusters (sometimes geographically distributed) and processed data locally in batches [27]. Today, the Fog and Edge computing paradigms are emerging as the best enablers for DML algorithms as they provide opportunities for offloading centralized computations by discovering processing capabilities in close proximity to the data sources [7]. In employing these DML methods, data-protection was raised as one of the major concerns and was even more severe in large-scale data analytic applications [22], [25]. As a result, privacy-preserved DML solutions such as FL [11] and parameter server-based DML [28], [29], blockchain-enabled ML [30] and split learning [10] have been developed. When there is a trusted third party which can serve as a central service unit, FL and parameter server-based approaches were recommended for DML. Other alternative is the use of blockchain but discovering resource providers is the most critical challenge.

2.2 Federated Learning (FL)

FL is a collaborative ML (and also a distributed optimization) concept [31] that was developed by Google researchers to train distributed datasets in a centralized parameter server [28]. A shared predictive model is collaboratively trained without moving data away from the participating client nodes such as mobiles and gateway devices. Hence, this approach has a greater potential in using a wider range applications where there is a need in performing timely analytics by incorporating widely distributed data. However, there are critical concerns regarding sharing data due to ownership, privacy and computational resources. To minimize the criticality of these concerns, FL process mainly consists of three steps as follows (also see Figure 1):

1. Each client trains (updates) the model, i.e., computes the model parameters by using its own data, and then transmits updated models (local models) to the central service unit.
2. The central service (coordination) unit (e.g., parameter server) collects the local models (i.e., local model parameters) and then aggregates them to compute the updated global model parameters. That is, every

client service contributes collaboratively to train a common ML model.

3. Every client retrieves global model parameters from the central service unit and uses them for making their own decisions and also, for the next model updating cycle.

These three steps repeatedly execute as FL progresses.

Gradient Descent (GD) algorithms have commonly been used for computing model parameters [32]. Assume a FL model that needs to update a parameter matrix of $W^{d_1 \times d_2}$. The i^{th} client service (assume C number of client services in total) will download the parameters at time t, W_t^i, and calculate the gradient $H^i = W^i - W$. Then, $W_{t+1} = W_t + \eta H_t$, where η is the learning rate. According to the averaging principle, $H_t = \frac{1}{n} \sum H_t$. There are some variants of GD algorithms such as mini-batch GD and Stochastic GD(SGD) [33]. In SGD, a single training sample is used at a time to train the model while mini-batch GD uses a small subset of the training dataset. Both have been proposed to overcome the processing burden in deep learning algorithms using image like data with a vast number of parameters in the order of 1000s or more. For instance, structured- and sketched-updating are two FL approaches that commonly use the SGD for distributed training of a ML model particularly optimizing the up-link communication for sending the updated parameters to the server when the resources are restricted [31].

The quality of the federated ML model and the optimization technique, which is used to train the model, are as crucial as the quality of the data used. Since a CML model is transformed into a set of small devices, each using a subset of a large training dataset, this model achieves data locality and harnesses the computational power of distributed edge/fog computing. Also, FL makes use of idle processing power by facilitating a certain number of clients to act as model trainers. Since the datasets are located in the client's devices, FL has great potential in reducing communication cost and preserving the data ownership and privacy issues. Recent research [21], which used FL with the support vector machine classifier for predicting heart diseases, and preserves the privacy of patients data, is a good evidence to emphasize the significance of FL. Moreover, the reasons to use FL as an alternative to current ML techniques with privacy preserving were broadly discussed in [11], while highlighting some common issues of FL, such as data imbalance and misbehaviour (or failure) of the central service unit, which requires careful attention when selecting it.

2.3 Involvement of Deep Learning in modern ML

The involvement of deep learning techniques such as convolutional, recursive, and recurrent NNs in advanced ML research like image recognition, object detection, and video analysis [12], [34] has intensified in recent years. Consequently, FL has also been influenced by deep learning techniques. For instance, the study in [32] used a CNN model to study the communication efficiency of their ML model under FL settings. The local Stochastic Gradient Descent (SGD) optimization technique based *Federated Averaging* algorithm proposed in [32] proved that the ML model could train with minimal communication requirements and that could help to overcome the communication issues in FL [11]. However, the applicability of deep learning models was limited particularly in many resource-limited infrastructures such as sensor-based analytics.

As an alternative, hybrid ML models, i.e., the combined use of deep learning models with conventional ML models, were developed. For instance, the study [13] proved that a combined use of NN and PLSR has the potential to generate more robust outcomes as well as optimize the resource consumption compared to using the algorithms separately. Similarly, different versions of hybrid models such as PCA combined with NN (PCANet) [35], PLSR with NN (PLSNet) [36], CNN based PLSR (CNN-PLS or stacked PLS) [34], and Hybrid-DBSCAN model to optimize clustering throughput in GPU and CPU [37] have been proposed and applied in various applications. Some of the applications include failure diagnosis of railway infrastructures [13], traffic incident detection [12], inland water quality evaluation [38], and image analysis [39]. Use of hybrid models is always limited to deep learning applications and has been realized in many other purposes. For example, [40] proposed a hybrid mathematical formulation for optimizing the computational energy required at data centers intending to reduce the ecological impact arising from data processing (i.e., Greenhouse gas emission). Also, [41] proposed a hybrid energy harvesting system and proved that the proposed system has better performances when compared to a single source harvesting approach. However, there is no evidence that such hybrid models have been used under DML settings, and in particular for FL framework.

2.4 Smart Dairy Farming

With the growing adoption of modern technologies such as milking robots and remote sensing in smart farming, more data-driven and data-enabled services are available today. Timely recommendations and relevant management strategies based on analyzing the information collected play a crucial role in accelerating the sustainable intensification of food production while optimizing resource utilization [22], [42]. This is a major requirement to address the challenges that will result from limited land availability, high labour cost, as well as climate change, aiming to support food demand for the 9 billion world population by 2050 [43]. This not only emphasises the need for effective computing services equipped with ML mechanisms that are trained dynamically, but have the capacity to be distributed and cooperative [42], [44]. However, most farms operate in isolation, which in turn, limits their interoperability. Resource constraints, functional incompatibility of the existing IoT platforms hamper such analytics. Besides these limitations, farmers are reluctant to share their data due to privacy and ownership issues. In this context, FL can provide effective services for deriving decisions by integrating insights extracted from a large number of distributed data sources.

In general, FL has been beneficial for supporting services for different applications and particularly where data privacy, constraints in resources, and ownership becomes major concerns in performing data analytics. At the same time, the use of hybrid ML models have received considerable attention for a wide range of analytical applications.

The importance of FL, however, has not been thoroughly realized yet in smart agriculture where communication and computational resources and also data-privacy are significant issues.

3 Federated Learning with a NN-based Partial Least Square Regression

This section, first, briefly describes the NN and PLSR methods and then explains how these two methods are combined to derive the joint ML model known as NNPLS. Next, the two approaches to federate the derived joint models are discussed. Finally, the evaluation metrics used for assessing the FL performances are described.

3.1 Partial Least Square Regression (PLSR)

Least square regression (LSQR) fails when the predictor variables are strongly correlated with each other and the number of informative features is larger than the number of data points. PLSR is a projection method [45], and considers not only the correlations between the predictor variables (X), but also the correlations among the predictor and the response (y) variables. By doing so, PLSR overcomes the limitations of the LSQR method, transforming the dataset into a lower dimensional space (latent space), where the LSQR can be used. Therefore, the general procedure of PLSR consists of two steps; dimension reduction and the application of LSQR, and are listed as follows:

1. The PCA technique is used for the dimension reduction by deriving the PLS factors (or Latent Variables (LVs)), which explain most of the variation in X and y.
   That is, PCA decomposes X and y using the singular value decomposition method as: $X = G_X P_X^T$ and $y = G_y P_y^T$, where G and P represents the score and loading matrices (their subscript stands for the matrix which they are derived from), respectively. Suppose q LVs are selected, where normally cross-validation technique is used, the X can be represented as:
   $$X = g_{X,1}p_{X,1}^T + g_{X,2}p_{X,2}^T + \cdots + g_{X,q}p_{X,q}^T + E_{X,q}, \quad (1)$$
   where $\{g_{X,i}\}_{i=1}^q \in G_X$, $\{p_{X,i}^T\}_{i=1}^q \in P_X$, and $E_{X,q}$ is the error matrix when the first k LVs are used to form the PLSR model.
2. LSQR is used to derive the PLSR model as follows,
   $$Y = p_{Y,1}g_{X,1} + p_{Y,2}g_{X,2} + \cdots + p_{Y,q}g_{X,q} + E_{Y,q} \quad (2)$$
   where $\{p_{Y,i}\}_{i=1}^q \in P_Y^T$, $\{g_{X,i}\}_{i=1}^q \in G_X$ and $E_{Y,q}$ is the error vector.

Further details on the PLSR process can be found in [45].

3.2 Neural Network (NN)

Suppose a simple NN with n nodes in the input layer with a single hidden layer, which consists of l nodes, and the output layer has only one node (see Figure 2) and named as $(n, l, 1)$ NN model. $\{W_{n\times l}^h, b_{1\times l}^h\}$ and $\{W_{1\times l}^o, b^o\}$ are respectively the input and hidden layer weight (W) matrices, including the bias (b) term. The incoming and

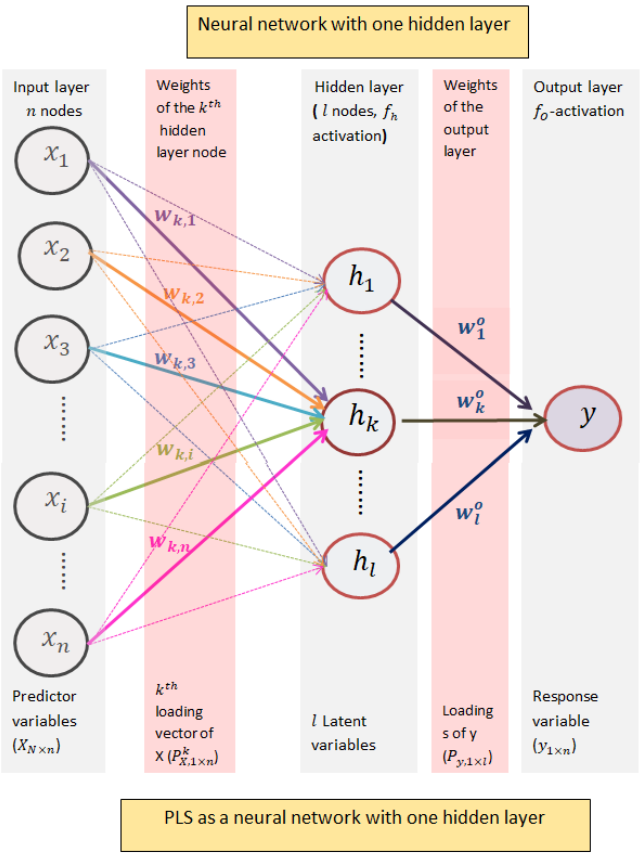


Fig. 2. PLSR model as a naive NN with one hidden layer. The top and bottom of each vertical color bar explains the entities of the NN and PLSR models, respectively. The weights in the hidden layer corresponds to the k^{th} node/LV.

the outgoing information, denoted as h_{in}^t and h_{out}^t of the k^{th} hidden layer nodes, are computed as follows (Figure 2 provides the description of the process at the top of each colored bars):

$$\begin{aligned} h_k^{in} &= w_{k,0} + x_1 w_{k,1}^h + x_2 w_{k,2}^h + \cdots + x_n w_{k,n}^h \quad (3) \\ &= \sum_{i=0}^{n} x_i w_{k,i}^h \\ h_k^{out} &= f_h(h_k^{in}), \quad (4) \end{aligned}$$

where f_h is the hidden layer activation function, $\{w_{k,i}\}_{i=1}^l$ is the k^{th} column of the matrix W^h, and $w_{k,0}$ is the k^{th} element of the vector b^h. The input data sample is represented as $x_1, x_2, \cdots, x_n \in X_{N\times n}$.
The same procedure is repeated for all other nodes in order to compute their outputs that will represent the inputs for the output layer (y_{in}). Based on this, NN calculates the output (y_{out}) as follows:

$$\begin{aligned} y_{in} &= w_0^o + h_1^{out} w_1^o + h_2^{out} w_2^o + \cdots + h_l^{out} w_l^o \quad (5) \\ &= \sum_{i=0}^{l} h_i^{out} w_i^o \\ y_{out} &= f_o(y_{in}) \quad (6) \end{aligned}$$

where $w_0^o = b^o$ and $\{w_i^0\}_{i=1}^l \in W^o$ are the output layer bias and weight matrix, respectively, and f_o is the output layer activation function.
In order to improve the robustness of y_{out}, NN optimizes

the weights as well as the bias by minimizing the errors between the actual y and y_{out} during the training process. The most extensively used training algorithm is the back-propagation and the process explained above presents the forward propagation step as it required for explaining the process of deriving the joint model only. Whereas, the back-propagation technique is used for training the joint model. There are other different optimization techniques for learning rules such as SGD. $f(x) = x$ (*linear*), $f(x) = \frac{1}{1+e^{-x}}$ (*Sigmoid*) are some of the frequently used activation functions.

3.3 NNPLS Model

When the number of hidden nodes is equal to the number of LVs (i.e., $l = q$), and the input and output layer weight matrices (i.e., W^h and W^o) is equal to the score and loading matrix of X (G_X^T) and y (P_y^T), respectively, the computations represented by equation 3 and 5 are respectively equal to the equation 1 and 2. In other words, one forward propagation step of the $(n, l, 1)$NN model given in Figure2 is equal to the PLSR model with $q(= l)$ number of latent variable. On the other hand, the PLSR model can be considered as a NN model with one hidden layer (number of hidden layer nodes is equal to the number of latent variables). The single node in the output layer is illustrated in Fig 2. This is the basic concept behind the joint PLSR and NN techniques for deriving the NNPLS model. Then the derived NNPLS model is trained in four steps, which are as follows (see Figure3):

1. Apply suitable pre-processing on a given dataset $[X, Y]$ such as PCA, scaling, and centering.
2. The optimal number of hidden nodes required for the NNPLS model is the number of LVs, which is derived from the PLSR-based cross-validation technique.
    a. Different PLSR models are fitted to the data by varying the number of LVs. In each fitting, the cross-validation error ($RMSE_{CV}$ - explained later) is computed by repeating the 10-Fold CV for 10^3 iterations.
    b. The number of LVs corresponds to the minimum $RMSE_{CV}$ and is selected as the optimal number of LVs, which is also the number of hidden nodes (l) for the NNPLS model. For instance, in order to get an idea about selecting optimal LVs, Figure 4 represents the behavior of $RMSE_{CV}$ with respect to the LVs for Fat milk percentage using the MIRS dataset discussed in Section 4.1. As observed in the graph, the minimum $RMSE_{CV}$ occurs at 6 LVs, which is the number of hidden nodes required for NNPLS model.
3. The score matrix of X and the loading matrix of Y that correspond to the optimal LVs are taken as the initial weights of the input and output layers that are required for starting the NN training process with selected activation functions and the optimization technique. The *rectified* and *linear* activation functions were respectively used for f_h and f_o. The optimization technique used was the $ADAM$ method.
4. Perform model updating based on a preferred approach and here we explain the sequential and parallel updating under FL settings in the next section.

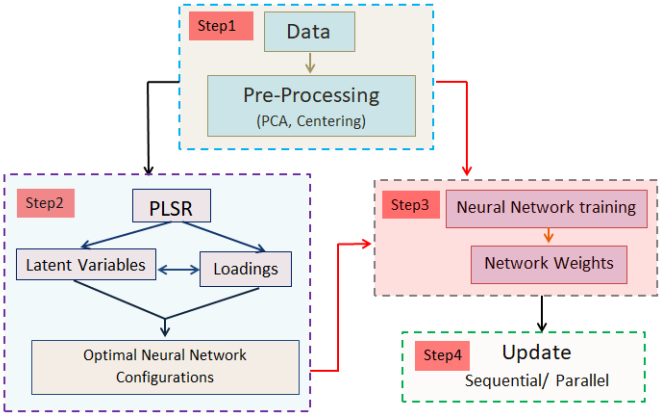


Fig. 3. Workflow of the NNPLS model for data pre-processing and model development steps.

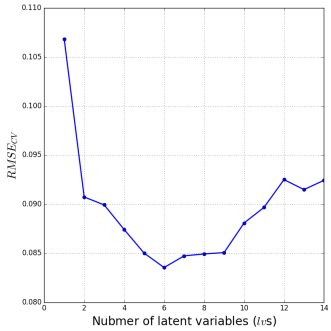


Fig. 4. Determination of the optimal number of LVs (i.e., hidden layer nodes) by using the MIRS dataset explained in Section 4.1 with Fat milk quality parameter; the optimal value of LV corresponds to the minimum $RMSE_{CV}$.

3.4 NNPLS model based Federated Learning (FL-NNPLS) architecture

In the NNPLS model based FL, the set of parameters that has to be federated is the NN weights derived from each client located in distributed locations. Each client service trains a common NNPLS model by using the weights downloaded from the central service unit. After training, the client service will send back the updated weights to the central service unit. The central service unit aggregates them and computes the final updated weights, and sends back to every client unit in order for them to update their models and perform predictions. In this process we assume that all client services contribute to updating the model at each federation step as well as accepting the final model updates. This means that the NNPLS model is dynamically updated based on the new datasets collected from each client over time. This is the core functionality of the FL-NNPLS architecture, which updates the model sequentially as well as in parallel.

3.4.1 *Parallel Updates*

The process for parallel updates follows the three steps as illustrated in Figure1, and is as follows:

1. Each client service independently trains a common NNPLS model using the available data.
2. The central service unit collects the NNPLS model weights sent by each client then averages it in order to get the final updated global weights
3. Each client service unit downloads the global weights and updates the model to perform predictions accordingly.

In the next federating step, which is the updating process, the global weights from the previous federating step are combined with the new loading metrics derived from the PLSR method by using the new dataset. This is then used as initial weights for training the NNPLS model.

Let us assume there are C client service units. Then at the $(t-1)^{th}$ federating step, the final input and output NNPLS model weights are denoted as W_{in}^{t-1} and W_{out}^{t-1}, respectively, and are derived by averaging the weights received from the C client services as follows:

$$W_{in}^{t-1} = \frac{1}{C}\sum_{i=1}^{C} W_{in,i}^{t-1} \qquad W_{out}^{t-1} = \frac{1}{C}\sum_{i=1}^{C} W_{out,i}^{t-1}$$

where $W_{in,i}^{t-1}$ and $W_{out,i}^{t-1}$ are the input and output NNPLS model weight matrices of the i^{th} client.
These weights are then sent back to all the client services to assess their model performance and is also used for computing the initial weights for the next federation step (t^{th}). At the t^{th} step, the i^{th} client service applies the PLSR to its new dataset $\{X_i^t, Y_i^t\}$ and derives the score and loading matrices of X_i and Y_i respectively as $G_{X,i}^t$ and $(P_{Y,i}^T)^t$. Following this process, the initial weights for training the NNPLS model are computed by averaging these loading matrices with the final model weights from the $(t-1)^{th}$ step, which is represented as follows:

$$W_{in,i}^{t,init} = \frac{G_{X,i}^t + W_{in}^{t-1}}{2}, \quad W_{out,i}^{t,init} = \frac{(P_{Y,i}^T)^t + W_{out}^{t-1}}{2}$$

This computation is performed for $i = 1, \cdots, C$ and aggregated in order to obtain the updated weights that are used to evaluate the FL performances at the t^{th} federation step, assuming $W_{in}^0 = 0$ and $W_{out}^0 = 0$.

3.4.2 Sequential Updates

In the case of sequential updates, the process is performed in a sequential manner. Three steps of the updating process are as follows:

1. The NNPLS model training process starts at a randomly selected client service, assuming that this client has sufficient data to start the training process.
2. The central service unit collects the model weights sent by that client service.
3. The next client service unit, which is ready to perform the model training process, downloads the weights from the central service unit and update its NNPLS model.

This process continues sequentially and one federating step is completed when the model at all the client services are up-to-date. The weights from the client service that is used for performing the last training is used as the final weights of the FL-NNPLS system, which will be utilized for the next federation step.

Let us assume the final NNPLS model weights from the $(t-1)^{th}$ federation step are W_{in}^{t-1} and W_{out}^{t-1}. At the t^{th} federation step, the first model training is performed by the i^{th} client, where $i \in \{1, \cdots, C\}$. The initial weights are computed as follows:

$$W_{in,i}^{t,init} = \frac{G_{X,i}^t + W_{in}^{t-1}}{2}, \quad W_{out,i}^{t,init} = \frac{(P_{Y,i}^T)^t + W_{out}^{t-1}}{2}$$

where $G_{X,i}^t$ and $(P_{Y,i}^T)^t$ are respectively the score and loading matrices of X_i^t and Y_i^t of the i^{th} client service at the t^{th} federation step. The NNPLS model is then trained to compute the final weights. The t^{th} federation step is completed once all client services have finished updating the model. This procedure is continued for $t = 1, \cdots, T$ assuming that at $t = 1$, $W_{in}^0 = 0$ and $W_{out}^0 = 0$.

3.5 Evaluation Metrics

The metrics, Residual Sum of Square (RSS) and coefficient of determination (R^2) are used to evaluate the predictive accuracy of the LSQR, PLSR, and NNPLS models under the federated and non-federated (i.e., CML) settings.

The RSS quantifies the square sum of the residuals and computed as:

$$RSS = \sum_{i=1}^{N}(y_i - \hat{y}_i)^2,$$

where y and $\hat{y}$ are actual and the predicted response variables and N is the sample size.

R^2 depicts the proportion of variance in the response variable y, which is related to the predictor variables in X. Therefore, we use R^2 as the accuracy measure to represent how accurate a ML model can predict a response variable y in our evaluations and compute as:

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2},$$

where $y, \hat{y}, N$ have the similar meanings as explained under $RMSE$.

4 Performance Evaluations

In this section, we explore the NNPLS model performances compared to the ML models of LSQR and PLSR and also, compare the learning performances of the NNPLS model based FL and non-FL (i.e., CML) settings. To perform these experiments, we use a dataset of MIRS from bovin milk. Initially, the dataset is briefly introduced, including the pre-processing steps. Based on the predictive performance obtained using a LSQR model, we explain the different characteristics of FL. This is followed by a discussion on the non-federated CML performance of the NNPLS model relative to LSQR and PLSR models using three different predictive parameters. Finally, we examine the performance of the FL-NNPLS approach for the same predictive parameters, and also compares them with the existing FL approaches.

Although our MIRS data explained below collected from different farms, it is available as a single dataset without

any information about the locations (i.e., farms) where the data samples were collected. So, in order to prepare it for validating the proposed approach, the following process was used to create a distributed data environment. When the dataset is used for distributed learning (i.e., training the NNPLS model under FL settings), we divide the MIRS dataset among five distributed clients. The number of clients was limited to 5 in order to have a sufficient number of samples per client (means client service) to perform the learning. This is because the original MIRS dataset consists of 712 samples only (each subset has $712/5 \approx 140$ samples). In a real-world scenario, this is feasible because in general, the herd size of an average dairy farms is around 100-150. The sub-samples are collected from client services into a central location (e.g., Cloud infrastructure) when performing CML. This is assuming no privacy concerns are raised for communicating data. In each analysis task explained below, 80% of the total samples were randomly selected for model training and the remaining samples were used to texting. Also, the learning performance presented below is also correspond to the learning accuracy derived from the test dataset.

4.1 MIRS Dataset and Pre-processing

The data used in this study originated from the Teagasc research dairy farm at Moorepark, Ireland where MIR spectra of milk were collected. The composition of milk was determined using the FOSS MilkScan [46]. The dataset consists of MIR spectra of 712 different milk samples in the wavenumber region $925 - 5005cm^{-1}$ with a resolution of $3.853cm^{-1}$. The wavenumbers were rounded to the nearest integer. As a result, the given spectrum contained 1060 transmittance data points. Hence, the original gold standard MIRS spectra used for FL was a 712×1060 size matrix. We converted them to absorbance values by taking log_{10} from the reciprocal of the transmittance values. The absorbance values in the milk samples indicate the amount of absorption of the electromagnetic radiation when the MIR light penetrates through the milk sample. Higher absorbance values indicate that the MIR light penetrates less at certain wavenumbers according to the molecular bonds. In addition, the percentages of the selected milk nutrient components (MQTs), Lactose, Fat, and Protein, corresponding to each milk sample were stored in a matrix ($y_{n \times k}$, where $n = 712$ and $k = 3$).

In spectrometry-based data analysis, pre-treatments are necessary as MIRS data contains large quantity of redundant data which adds variability in the wavelengths. Also, the higher dimensionality and multi-collinearity among the wavelengths limits the use of simple ML models. Consequently, these factors could affect the resulting predictive accuracy. The original milk spectrum indicated two random sharp fluctuation regions, which occurs in the wave number regions $1500 - 1800cm^{-1}$ and $2900 - 3800cm^{-1}$ per visual observation (see Figure 5). These regions are the water absorbance regions according to the pure water spectrum at $25°C$, which corresponds to $O = H$ bonds in the spectrum. We identified these two regions based on PLSR model calibration, which was conducted on our gold standard data and removed it in the pre-processing stage. Then the water absorbance regions removed were $1607 - 1734cm^{-1}$ and $3021 - 3707cm^{-1}$ [18], [19]. By removing these two regions, the dimensionality of water free spectrum (X) was 712×847. Figure 5 represents the water absorbance regions removed spectra.

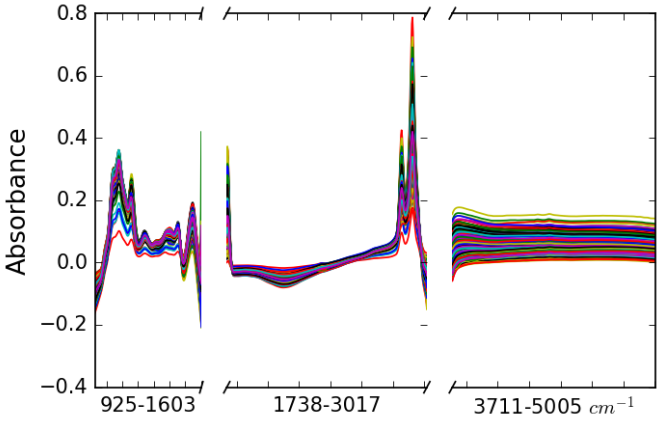


Fig. 5. Water absorbance regions removed MIR spectra of 712 milk samples within the wave number region $2500 - 25000nm (900 - 5000cm^{-1})$ of the electromagnetic spectrum.

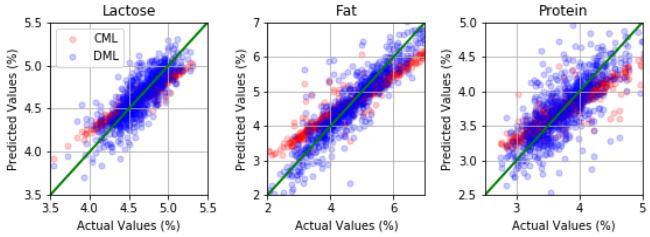


Fig. 6. Federated and centralized prediction accuracy of MQTs in the MIRS dataset (DML with 5 clients).

Scaling MIRS data was not a compulsory approach since all the features were in the units of absorptions. Therefore, the water-removed MIRS data was then fed into PCA dimension reduction stage and the PCs corresponding to the reconstruction error less than 10^{-4} were selected. Reconstruction-error is the l_2 norm of $(X - \hat{X})$, where $\hat{X}$ is the reconstructed X by PCA. These pre-processing steps could precisely remove the wavenumbers from the original spectra to obtain pre-processed MIR spectra (say X) for use in FL. We used Python scikit-learn and TensorFlow libraries for all the analytical work.

4.2 FL performance of MIRS Data with a LSQR model

A LSQR model was formed including all the parameters in X (i.e., PCs with reconstruction error $\leq 10^{-4}$) and then federated once by equally distributing the 712 samples among the 5 clients (≈140 samples per client). With a 10^{-3} learning rate, the SGD algorithm was then used to train the model at each client for 10^3 iterations. The predictive performance of the FL (i.e., DML) approach (parallel) was compared with the non-FL (CML) approach. Figure 6 shows the accuracy of CML and DML approaches for three MQTs. The predictive and actual MQT values under the DML approach are distributed around the straight line (i.e., actual y and predicted y ($\hat{y}$) should be linearly correlated as $y = m\hat{y}$, where $m = 1$) compared to those under the CML approach. That is, the DML (FL) predictive performance is better than that of the CML with LSQR model.

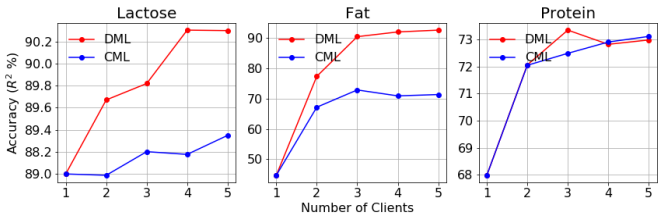


Fig. 7. Comparison of FL and CML performances of training efficiency with number of clients obtained using MQTs.

Following the similar procedure used above, the variability in predictive performance of FL with increasing number of clients was studied with the results summarized in Figure 7. The FL approach achieved higher accuracy and converged faster than the CML approach. Therefore, having many clients contributes to improving the FL efficiency by speeding up the model training process. The effectiveness of the FL approach mainly depends on the number of clients as well as the number of samples used by each client to train the ML model. Having a set of suitable initial model parameter values generally guarantees better accuracy when compared to their optimal values. So the updated set of model parameters that are available at the central processing unit is sent to the next client(s) as initial parameter values. The convergence of SGD gradually becomes faster as federation progresses.

These results confirm that FL performs better than the CML approach, but the performance can be improved further with increasing number of clients as well as using larger training samples. However, the limitations of LSQR mentioned in Section 3.1 might cause a poor predictive performances for some MQTs [18]. The next section explains how to overcome those limitations based on the computational evidence derived from using PLSR and NNPLS models.

4.3 Centralized and FL performances with NNPLS model

In this section, non-federated (CML) performance of the NNPLS model is compared to the LSQR and PLSR models. Then the performance of FL-NNPLS is explored.

Prior to exploring learning performance, in order to guarantee the convergence of the optimization algorithm (SGD), we experimented variability in error loss (as in RSS Loss - Residual Sum of Square Loss) with increasing number of clients under both the Federated and Non-Federated settings. For instance, Figure 8 depicts the convergence of RSS Loss up to three clients with the NNPLS model for 10^3 iterations only. It can be seen that the convergence is faster with more clients under federated settings than the centralized settings.

Table 1 represents the non-FL predictive accuracy obtained from the LSQR, PLSR, and NNPLS models for each MQT, including the number of LVs. The CNN model is explained in the next section. The replacement of the LSQR model by the PLSR and then the NNPLS models contributed to improve the predictive accuracy of all MQTs; the largest improvement was for Protein, followed by Lactose. This is highly likely attributed to the PLSR and NNPLS models taking into account the multi-collinearity in the MIRS data,

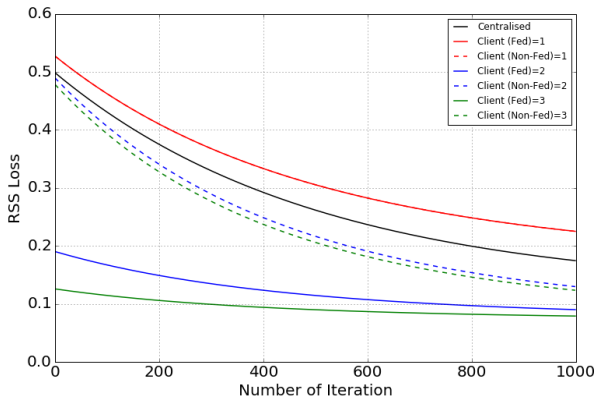


Fig. 8. Convergence of the optimization method (i.e., SGD) under non-Federated (i.e., CML) and Federated settings with the NNPLS model (the convergence was quantified by using RSS Loss - Residual Sum of Square Loss).

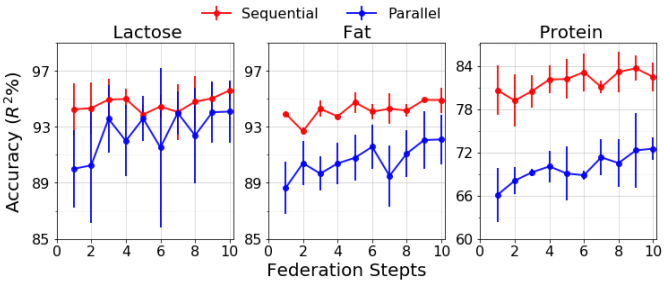


Fig. 9. NNPLS model-based FL performance for different MQTs under the sequential and parallel model updating approaches.

which is not considered in the LSQR model. Also, the NNPLS model is capable of capturing non-linear functional relationships in addition to multi-collinearity in MIRS data. Moreover, NNPSL model follows the SGD optimization to select the optimal parameter values, which is not considered in the PLSR model. That is why the accuracy of each MQT obtained from the NNPLS model is better to that of the LSQR and PLSR models. Thus, the NNPLS model is computationally more effective compared to the traditional NN models and also able to provide more precise learning outcomes compared to LSQR and PLSR.

With five clients, the NNPLS model of each MQT was federated for 10 times under the sequential and parallel updating approaches separately, assigning 140 ($\approx 712/5$) samples randomly for every client at each federation. This approach is used in the same training and validation settings, which were used under the non-FL approach. Figure 9 depicts the variability in the average predictive accuracy over the five clients obtained from both updating procedures at each federation step. In general, with increasing number of federation steps, the predictive accuracy of all MQTs increased in both updating approaches. The accuracy obtained from the sequential updating was, however, higher than the parallel updating approach. Furthermore, after ten federation steps, all MQTs achieved higher accuracy compared to the non-FL accuracy given in Table 1 for both updating approaches.

In general, the NNPLS model performed better than the LSQR and PLSR models with MIRS milk data. At the same time, the FL-NNPLS approach achieved greater per-

MQT	LSQR (%)		LVs	PLSR (%)		NNPLS (%)		CNN(%)	
	Train:	Vali:		Train:	Vali:	Train:	Vali:	Train:	Vali:
Lactose	92.43	91.62	12	94.05	90.86	96.41	93.39	97.85	92.58
Fat	93.49	87.50	5	93.95	91.44	96.11	91.12	96.36	92.21
Protein	83.86	66.87	5	75.62	69.44	87.46	83.09	82.05	77.66

TABLE 1
Centralized (non-FL) training (Train:) and validation (Vali:) accuracy ($R^2\%$) of LSQR, PLSR, NNPLS, and CNN models for different milk quality parameters (MQTs).

	FedAvg	FL-NNPLS
1). Clients contributes to global model derivation	Subset	All
2). Local model aggregation for computing global model	Weighted mean	Arithmetic mean
3). Global model updating methods	Parallel	Parallel and Sequential
4). Local model optimization methods	SGD	SGD + (PCA + PLS + NN)

TABLE 2
Comparison of properties between FedAvg and FL-NNPLS methods.

formance compared to the performance obtained from the CML approach. Moreover, the sequential updating based FL performance has better performance compared to the parallel updating based FL approach.

Moreover, we explored the learning capability of the FL-NNPLS method and compared with already existing FL methods Federated-Averaging (FedAvg) and Federated Stochastic Variance Reduced Gradient (FSVRG). In terms of global model updating procedure, the FedAvg approach is similar to the FL-NNPLS under the parallel updating method discussed in section 3.4.1. Whereas, the local model aggregation method at the central service unit (or server) is different. The FL-NNPLS uses the arithmetic mean while the FedAvg uses the weighted mean for aggregating local models. That is because, in the present study, since we assumed that all clients have equal sized datasets, the arithmetic mean was used to aggregate local model updates for computing the global model. However, FedAvg does not necessarily make the same assumption. Because FedAvg uses only a set of clients to update the global model updating process though FL-NNPLS uses the model updates from all clients to compute the global model. This in turn makes the FedAvg particularly useful for application where there is a variability in data generation speed among clients. In addition, the FL-NNPLS uses SGD along with PCA, PLSR and NN methods for computing local model updates, but FedAvg mostly uses SGD only. So the chance of achieving greater performance from the FL-NNPLS method may be relatively higher than the FedAvg. It may, however, vary depending on application-specific requirements such as availability of data and computing resources. Table 2 summarises these similarities and differences.

Considering the FSVRG method, the global model derivation procedure is also similar to the FL-NNPLS-based parallel model updating process discussed in section 3.4.1. Also, local model aggregation in the FSVRG method is similar to the FedAvg method. However, the main difference from both the FL-NNPLS and FedAvg methods is one full gradient computation is performed centrally, followed by many distributed stochastic updates over the distributed clients. Most importantly, the FSVRG is originally designed for taking into account spare data in the sense that detecting seldom features represented in local datasets through a data scaling procedure. Therefore, FSVRG could be an important method for applications where there is greater sparsity in local datasets. However, integrating models which have a large number of parameters with the FSVRG is challenging due to scaling issues that arise through mismatching between input and output model parameter dimensions. More details about the FedAvg and FSVRG can be found in [47].

Table 3 compares the predictive learning performance of FL-NNPLS (after 10 federation steps with 5 clients) with FedAvg and FSVRG methods using the above MIRS dataset (D-1) and another publicly available MIRS dataset (D-2) consists of 960 MIR milk samples. The difference between the FL-NNPLS and FedAvg approaches is that the FL-NNPLS takes the model updates from the all clients to compute the global model, while the FedAvg uses only a set of clients for updating the global model. Whereas, in the FSVRG method, one full gradient computation is performed centrally, followed by many distributed stochastic updates over the distributed clients. More details about the D-2 dataset can be found in [48]. So, to set up similar settings as the FL-NNPLS method (i.e., $712/5 \approx 140$ samples per client), with the the FedAvg method, total seven clients were considered ($960/7 \sim 140$ samples per client) and only 5 of them were contributed to the model updating process in each updating cycle. FL-NNPLS achieves comparable learning performance to the FedAvg and FSVRG with D-1 dataset though, it slightly less with the D-2 dataset. That is because the D-2 dataset contains almost 250 more MIRS samples than the D-1. Therefore, this comparison shows that the FL-NNPLS method has comparable performances to the already existing FL methods. Moreover, higher performances can be achieved with larger datasetes. However, lack of domain knowledge about other datasets (e.g., multicollinearity) and also diverse application specific requirements are some of the critical challenges that could be raised in applying the FL-NNPLS method for the analysis of data.

5 Discussion

The FL method can be considered as a realization of the concept of *Data Gravity* proposed by *Dave McCrory* in 2010. He pointed out that with increasing data sizes, the computational power should be shifted towards the data sources. Fog/edge computing and Cloudlets are two popular DML enablers where this concept is being practiced [19], [49]. In these distributed computing environments, FL is one of the latest optimization approaches which can be used to perform analytics. Therefore, in this section, first we discuss the state-of-the-art performances of the FL-NNPLS approach,

MQT	FL-NNPLS (%)		FedAvg (%)		FSVRG (%)	
	D-1	D-2	D-1	D-2	D-1	D-2
Lactose	95.61±0.63	95.98 ± 0.63	92.94± 0.12	94.47± 0.32	91.17± 0.54	96.45± 0.64
Fat	94.91±0.90	92.98 ± 2.87	85.23± 0.24	85.23± 0.43	86.04± 0.47	82.17± 0.17
Protein	82.44±1.96	85.62± 4.50	82.80± 0.25	69.44± 0.58	83.05± 0.54	88.79± 0.38

TABLE 3
Comparison of sequential updating-based learning accuracy (i.e., (mean ± std) of $R^2\%$) of FL-NNPLS, FedAvg and FSVRG obtained with 5 clients and 10 federation steps using datasets D-1 and D-2.

which uses Convolutional NN (CNN). The common issues of the DML approach are then discussed, including some of the challenges in FL-NNPLS technique. Subsequently, one of the critical challenges, which is data imbalance and finally, directions for future research under FL are discussed.

5.1 Comparison of FL-NNPLS model performance to a deep NN model

Considering different CNN models (e.g., LNet, Vgg-19, and Resnet), which are explained in [19], [50], [51], CNN model was selected based on LNet-5 and Vgg-19 models such that they are deeper (i.e., number of layers) than the LNet-5, but not as deeper as Vgg-19. This means that the CNN model consisted of three convolutional and three dense layers. Each convolutional layer contained a 3×3 sized kernel. The number of features extracted from each layer was 20, 30, and 40, respectively. Also, each convolution layer was followed by a max-pooling layer with a kernel of size 2×2. A flatten layer was included after the last convolution layer. Then the dropout layer dropped out by 20% neurons of the flatten layer. The first dense layer contained 30 neurons. The number of neurons in the second dense layer was equal to the number of LVs of the predictor variable used for learning. The last dense layer contained only one neuron and used the *linear* activation function. All other layers of the CNN model used the *rectified* activation function. *ADAM* optimization technique was used to train the CNN model. The reason behind selecting a CNN architecture between LNet-5 and Vgg-19 was that training large deep learning models like Vgg-19 under the FL settings may not be feasible under certain circumstances, such as limited resources and low complexity.

Under the CML settings, the water-free MIRS dataset was first compressed by applying PCA with 10^{-4} reconstruction error Then the compressed dataset fed into the CNN model. The model was trained for 10^3 times by selecting the initial network weights from the uniform distribution. The predictive accuracy for each MQT was then computed and given in Table 1. It is clear that the NNPLS model has state-of-the-art performance because the CNN model outcomes are comparable to the NNPLS model.

To train the CNN model under the FL settings, the same procedure which was used to train the NNPLS model in Section 4.3 was followed. However, the initial network weights were selected from the uniform distribution. If the square root of the selected number of PCs was not an integer, then the PCA compressed data could not reshape it to feed to the CNN model for training and validation. Therefore, the zero padding technique was used to adjust the number of feature variables in the compressed dataset before reshaping. The number of samples per client was increased up to 250, but

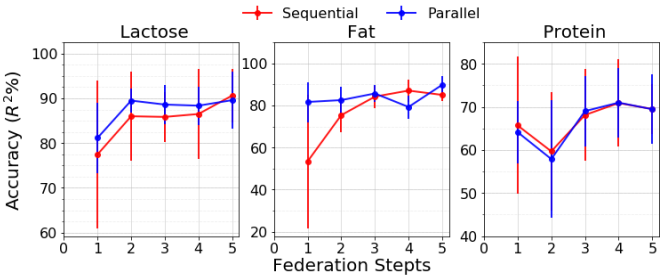


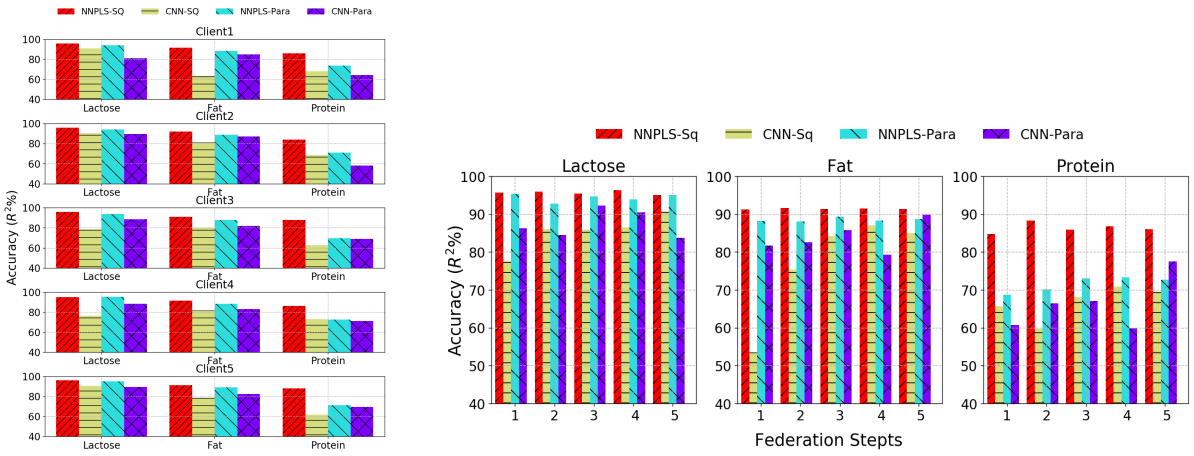
Fig. 10. CNN model-based FL performances under the sequential and parallel updating techniques.

the federation steps were limited to five as over-fitting is a common issue with the CNN model due to the small data size. Figure 10 represents the validation accuracy for each MQT with increasing number of federation steps under the sequential and parallel updating approaches. As shown in the results, the accuracy improved for all MQTs under two updating approaches.

Comparing the CNN-based FL performance to the FL-NNPLS outcomes, NNPLS model achieved higher accuracy for all MQTs compared to the CNN model under similar FL settings. Moreover, the convergence efficiency of NNPLS was faster than the CNN model, as CNN requires longer training time to achieve similar performance to FL-NNPLS. Therefore, these results prove that the NNPLS model results in greater performance compared to the CNN model under similar experimental settings. Figure 11 depicts the sequential and parallel updating performances based on the validation accuracy of each MQT obtained from the NNPLS and CNN models. While the variability in the predictive accuracy of all the MQTs at each client is displayed in Figure 11(a), Figure 11(b) represents the change in predictive accuracy for each MQT with each federation steps. In general, the predictive performance from the sequential updating method was higher than the parallel updating method for both models. Also, under the sequential updating technique, NNPLS model based predictive accuracy was comparable to that from the CNN model for all MQTs. However, considerable differences were observed for certain MQTs with the parallel updating approach. Therefore, it can be concluded that FL with the sequential updating performs well compared to the parallel updating. Also, the NNPLS model has comparable performance to the state-of-the-art CNN-based deep learning model based on the predictive performance obtained for the MIRS data of milk.

5.2 Advantages of FL-NNPLS

The DML framework based on FL-NNPLS approach can handle most of the issues mentioned heretofore. Since data



(a) Predictive accuracy of every MQT at each client.

(b) Variation of predictive accuracy with federation steps.

Fig. 11. Sequential and parallel updating based FL performances obtained from NNPLS and CNN models for three different MQTs.

will not migrate from the data sources, and ML model updates do not store them in the server, FL-NNPLS optimizes resource consumption as well as data privacy, security, and data ownership. Also, as every client integrates updated ML model immediately after each federation, they can make timely decisions effectively. In many applications, NNs are the state-of-the-art though selecting a proper network configuration is time and resource-consuming. However, modern deep learning models ResNet and GoogleNet can be easily optimized and gain higher accuracy with increasing depth of the models and can control the computational cost required with deep learning models [50]. Employing them in Fog/Edge computing based DML systems might not be possible particularly in smart farming, because such systems mostly rely on resource limited sensor nodes which cannot train those models. However, considering the applicability of modern DML frameworks such as *Horovod* [1] will be a good alternative to handle such limitations.

Since NN-based FL-NNPLS approach is associated with the PLSR technique in selecting suitable NN configurations, the learning process is faster and preserves the computational power. Having a proper set of initial network weights speeds up the network convergence, cuts down the computational burden required for deciding the NN configurations, provids a faster convergence rate. Since the PLSR technique provides these pre-requirements, NNPLS-based ML can provide computationally inexpensive, robust, and scalable solutions for a broader range of applications. Furthermore, PLSR effectively overcomes the multi-collinearity and higher dimensionality, while NN enables capturing complex functional relationships in the data. Therefore, the scalability of the FL-NNPLS approach is better compared to the NN or PLSR methods.

5.3 Data imbalance issue in FL-NNPLS

The ultimate purpose of this DML is for deriving meaningful and timely insights from massively distributed datasets. Hence the necessity for efficient FL frameworks has a growing demand with the growing prevalence of big data in a broad range of applications. Nevertheless, there are some common critical issues associated with it, and some are listed as follows:

1. The FL framework is used for learning from large-scale distributed data, but finding resources in order to respond to the ever-increasing data volume (e.g., online sensor data) is challenging.
2. The datasets involved in FL are typically heterogeneous and that brings up constraints such as aggregating the FL model parameters and defining a common representation for data to be able to apply ML. Also the datasets are not complete, balanced, and uncertain due to a number of reasons such as missing data or their unavailability, and the un-verifiability of all data sources. As a result, most ML algorithms cannot be applied directly so that deriving precise insights from such data could be challenging.
3. The FL system is totally dependent on the coordination device which provides services to the collection by the local model updates and using this to produce the global model updates. Any functional failure (or misbehaviour) of this entity could result in a collapse of the entire system.
4. It has been already warned that significant information can be extracted by tampering the model updates [52]. Hence, extra security efforts are now essential, particularly in smart farming applications for performing safe communication of model updates between the clients and the central service units.
5. The FL system is lacking a proper mechanism to examine the validity of the clients' data and model updates as they can inject false information into the FL system.

Therefore, FL systems should have the potential to understand these factors and be equipped with the necessary tools in order to efficiently overcome them. For example, to overcome the data imbalance issue in FL, we propose an approach by using the FL-NNPLS method and explore its performance based on the MIRS dataset.

Data imbalance is commonly encountered in many ML applications most notably in data classification and DML. This happens when the number of clients that have small

1. https://github.com/horovod/horovod

quantity of samples is significantly higher compared to the total number of clients. Consequently, ML outcomes are most likely biased towards the clients which have larger samples. ML algorithms such as data classification consider the clients are having a small number of samples (minority samples/classes) as noises and tends to neglect them during the learning process. In fact, minority classes are usually more critical. For instance, in dairy herds, there could be very few sick cows relative to the number of healthy cows, and this minority group plays a crucial role when identifying animals which are affected. Hence, data imbalance is an important topic in advanced ML research with the growing interest in DML. According to a recent study [53], there are two main approaches to overcome this issue; data-level and algorithm-level approaches. While the first approach overcomes data imbalance by using re-sampling techniques, different techniques such as adding penalty constants are used in the second approach to overcome the data imbalance issue. Many experimental attempts have proved that re-sampling, i.e., over-, under-, and hybrid-sampling, is a promising way to manage data imbalance [53]. However, the best-suited technique depends on the characteristics of the imbalanced datasets.

On the other hand, there is another problem associated with FL-NNPLS approach due to data imbalance; the number of LVs varies with the data size, and consequently the NNPLS model configurations vary over the clients and it hampers aggregation of model updates. That is, feature variables are heterogeneously distributed over clients and DML system aggregates different ML models in order to perform learning. In some studies, DML with different ML models has been named as vertically partitioned data or dimensionally distributed data [54]. However, these ML models that include the re-sampling methods cannot solve the imbalanced data issue alone in FL-NNPLS approach. Hence, a technique, which can control both issues at the same time, is required. Therefore, in the present study, a re-sampling and zero-padding based joint approach was used to overcome the issue.

Five clients were federated for 5 times under the sequential updating FL approach since it performed well compared to the parallel approach. In each federation step, the number of samples of every client was allowed to vary randomly between 50-250. PCA reconstruction error was set to 10^{-4} in order to select the optimal number of PCs. The same procedure was used in Section 3.4.2, but two additional steps were used here. The first one is to balance sample sizes of the clients. The random re-sampling was applied to the current FL participator only if its sample size was less than the previous client which performed model updating. Randomly selected samples from its own samples were used to balance the current sample size to the sample of the previous client (i.e., random up-sampling). The second step was added to the NN weight aggregation stage in which zero-padding was applied to equalize the sizes of the NN weight matrices. The variability in predictive accuracy was computed under both the CML and FL approaches.

Figure 12 represents the federated and non-federated predictive accuracy obtained for the milk Fat at each federation step for every client, including the original sample sizes and LVs. The predictive accuracy from the FL approach

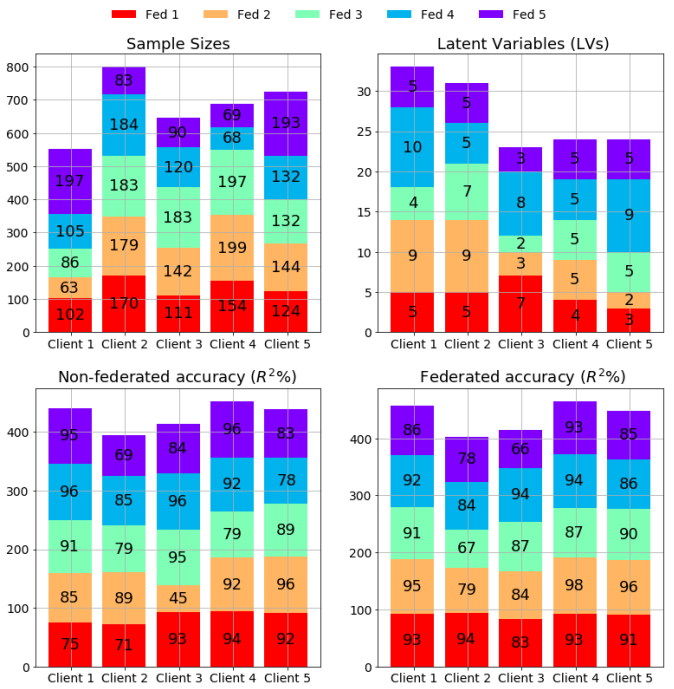


Fig. 12. The federated and non-federated NNPLS model performances with imbalance data.

is generally higher than the non-FL. Therefore, it seems that the re-sampling and zero padding based approach has the potential of mitigating data imbalance issues. However, further research is essential to study the validity of this approach in different applications.

5.4 Future Research Directions

Our future research directions can be considered in two ways; 1) finding solutions to overcome the challenges mentioned in Section 5.3 and 2) exploring novel approaches to enhance the performances of FL-NNPLS in distributed services.

Some of the possible solutions could be explored further to overcome these challenges are:

1. Exploring novel cooperative computing (resource sharing) approaches such as offloading computations to neighbouring devices with services as explained in [55] for minimizing the constraints in resources. Also, research in effective data compression techniques for compressing training data as well as communication over the FL system would be an interesting approach for minimising the computing and communication costs. In addition, incorporating the distributed cloud service model for resource allocation proposed in [56] and mobile edge computing approach presented in [49] would be better approaches to minimize resource limitations.
2. Designing techniques for FL with dynamically changing central service unit based on factors such as resource availability and communicability with the client's services would also contribute to improving the stability of the FL system, resulting in the minimization of the impact of failure (or misbehaviour) of the central service unit. The concepts about optimizing the node failure in connected devices given

in [57] would be beneficial exploring a path for mitigating the impact of central service unit failure in FL.

3. In order to prevent tampering the model updates of the services traversing over the FL system, public-private key based data encryption techniques such as the authenticated symmetric encryption with Diffie-Hellman (D-H) key exchange service can be incorporated with FL. Further exploration on how they facilitate to improve data privacy and communication resources will be required.
4. Proposing techniques for integrating FL with blockchain services, for validating model updates and also minimizing the misbehaviour of any FL participant.
5. Performing FL by selecting only a set of clients' services, who have good history of providing valid model updates, to update the final ML model will also be a promising solution to control the injection of false information to the FL system.

There are different ways for developing novel approaches to enhance the FL performance in distributed services.

1. One of the main goals of the FL approach is to achieve high accuracy with a minimum number of federated communication rounds. This goal can be achieved by using the ML algorithms which have faster convergence rates. Hence, developing a better theoretical understanding about the convergence properties of such algorithms will be an interesting future research direction.
2. Since FL is an optimization framework, a convex optimization function is essential to guarantee the convergence of the model parameters. However, NN-based ML yields non-convex function so there is no evidence regarding the guarantee of convergence of the optimization algorithm. Therefore, studies of the FL problems for non-convex objectives is another direction that may contribute to the efficient application of FL for solving even more complex problems in advanced applications such as object detection and image processing.
3. Developing a business model based on a token-based FL approach. This means that the clients who contributes to update the ML model can rent the model for outsiders who want the model for practicing different services such as making decisions. In return, they have to pay a certain amount of tokens, through a form of currency, to the FL client(s). This method will facilitate FL-based platforms to provide greater services to wider range of application domains.
4. The present study assessed the predictive performance of the NNPLS model based CML and DML only. However, exploring the computational efficiency would be an interesting extension of this work by taking into account various factors. Some of the factors, which could contribute towards improving the efficiency of the FL-NNPLS approach, are resource availability at the client services and central service unit, frequency of data collection, and model update transmission cost.
5. This study considered parallel and sequential updating processes by taking the whole parameter space in each client service unit, but in some cases, that may not be feasible. For instance, the available data at some clients may not be sufficient to update the whole parameter space, but only a part of it. In such cases, the parameter-server based approach presented in the study [29] incorporated with the approach presented to solve the data imbalance above could be an alternative as it enables updating the parameter space by partitioning it over a distributed worker groups. Therefore, extending this study with vertically partitioned parameter space over the distributed clients is another potential future study.

6 Conclusion

This paper presents the applicability and benefits of using a hybrid model of FL and ML models for distributed services. A particular application of the services is a case study on spectral data generated from milk samples, which essentially operates as a tool to predict three milk quality parameters. The NNPLS model developed for the FL model based on the limitations of the LSQR, PLSR, and NN models was used for predictions under the CML and DML settings. Under the CML settings, the NNPLS model contributed to improvement of the predictive performance compared to LSQR and PLSR models and also achieved comparable performances to the state-of-the-art CNN model. Therefore, the NNPLS model is a good fit for performing predictive analytics on milk quality data. Under the FL configurations, our NNPLS model achieved similar performance when compared to the CML approach, and by only using a few federation steps. Moreover, with the similar FL settings, FL performance of the NNPLS model was similar to the state-of-the-art CNN model. Therefore, FL-based NNPLS model can provide timely insights regarding the composition of milk while preserving data privacy and ownership with minimal resource requirements, which are critical challenges in providing effective services in modern day smart farming applications. The sequential updating based FL approach is a good fit for analyzing milk composition as it achieves better performance with both NNPSL and CNN models compared to the parallel updating approach. The re-sampling and zero-padding based approach contributed to mitigate the impact of data imbalance in FL. However, further investigation is required to improve the performance further.

Acknowledgment

This research was supported by the Science Foundation Ireland (SFI) project "PrecisionDairy (ID: 13/1A/1977)" as well as a research grant from Science Foundation Ireland and the Department of Agriculture, Food and Marine on behalf of the Government of Ireland under the Grant 16/RC/3835 (VistaMilk).

REFERENCES

[1] C. Kulatunga, L. Shalloo, W. Donnelly, E. Robson, and S. Ivanov, "Opportunistic wireless networking for smart dairy farming," *IEEE IT Professional Magazine*, vol. 19, no. 2, 2017.

[2] S. Capalbo, J. Antle, and C. Seavert, "Next generation data systems and knowledge products to support agricultural producers and science-based policy decision making," *Elsevier J. of Agricultural Systems*, vol. 155, pp. 191–199, 2016.

[3] S. Wang, L. Huang, Y. Nie, X. Zhang, P. Wang, H. Xu, and W. Yang, "Local differential private data aggregation for discrete distribution estimation," *IEEE Transactions on Parallel and Distributed Systems*, pp. 1–1, 2019.

[4] H. O. Alanazi, A. H. Abdullah, and K. N. Qureshi, "A critical review for developing accurate and dynamic predictive models using machine learning methods in medicine and health care," *Medical,Systems*, vol. 14, no. 4, 2017.

[5] "Department of Agriculture and Food and the Marine and Ireland, Food Wise 2025 - Local Roots Global Reach: A vision for growth for the Irish agricultural economy for the next 10 years," https://www.agriculture.gov.ie/foodwise2025/, 2018.

[6] J. Qiu, Q. Wu, G. Ding, Yuhua, and S. Feng, "A survey of machine learning for big data processing," *EURASIP J. of Advances in Signal Processing*, 2016.

[7] P. Sun, Y. Wen, T. N. B. Duong, and S. Yan, "Timed dataflow: Reducing communication overhead for distributed machine learning systems," in *IEEE Conf. on Parallel and Distributed Systems*, 2016.

[8] F. Jalali, K. Hinton, R. Ayre, T. Alpcan, and R. S. Tucker, "Fog computing may help to save energy in cloud computing," *IEEE J. of Selected Areas in Communications*, vol. 34, no. 5, 2016.

[9] E. P. Xing, Q. Ho, P. Xie, and D. Wei, "Strategies and principles of distributed machine learning on big data," *J. of Engineering*, vol. 2, pp. 179–195, 2016.

[10] R. Raskar, P. Vepakomma, A. Singh, O. Gupta, V. Pamplona, and K. Pho, " Distributed deep learning and inference without sharing raw data," http://splitlearning.github.io, 2018.

[11] J. Konecny, H. B. McMahan, D. Ramage, and P. Richtarik, "Federated Optimization: Distributed Machine Learning for On-Device Intelligence," https://arxiv.org/abs/1610.02527, 2016.

[12] J. Lu, S. Chen, W. Wang, and H. v. Zuylen, "A hybrid model of partial least squares and neural network for traffic incident detection," *Elsevier J. of Expert Systems with Applications*, vol. 39, pp. 4775–4784, 2012.

[13] A. Debiolles, L. Oukhellou, and P. Akrin, "Combined use of partial least square regression and neural network for diagnosis tasks," in *IEEE Conf. on Pattern Recognition (ICPR'04)*, 2004, pp. 1051–4651.

[14] G. Visentin, A. McDermott, S. McParland, D. P. Berry, O. A. Kenny, A. Brodkorb, M. A. Fenelon, and M. D. Marchi, "Prediction of bovine milk technological traits from mid-infrared spectroscopy analysis in dairy cows," *Elsevier J. of Dairy Science*, vol. 98, 2015.

[15] M. Taghdiri, G. Karim, S. Safi, A. R. Foroushani, and A. Motalebi, "Study on the accuracy of milk amyloid a test and other diagnostic methods for identification of milk quality," *Vet Research Forum*, vol. 9, no. 2, pp. 179–185, 2018.

[16] S. McParland and D. P. Berry, "The potential of fourier transform infrared spectroscopy of milk samples to predict energy intake and efficiency in dairy cows," *Elsevier J. of Dairy Science*, vol. 99, 2016.

[17] L. Hedlund and H. Lovlie, "Personality and production: Nervous cows produce less milk," *Elsevier J. of Dairy Science*, vol. 98, no. 9, pp. 5819–5828, 2015.

[18] D. Vimalajeewa, D. P. Berry, E. Robson, and C. Kulatunga, "Evaluation of non-linearity in mir spectroscopic data for compressed learning," in *IEEE ICDM Workshop on High Dimensional Data Mining (ICDM-HDM)*, 2017.

[19] D. Vimalajeewa, C. Kulatunga, and D. P. Berry, "Learning in the compressed data domain: Application to milk quality prediction," *Elsevier J. of Information Sciences*, vol. 459, pp. 149–167, 2018.

[20] J. Kang, R. Yu, X. Huang, S. Maharjan, Y. Zhang, and E. Hossain, "Enabling localized peer-to-peer electricity trading among plug-in hybrid electric vehicles using consortium blockchains," *IEEE Trans. on Industrial Informatics*, vol. 13, no. 6, pp. 3154–3164, 2017.

[21] T. S. Brisimia, R. Chena, T. Melac, A. Olshevskya, I. C. Paschalidisa, , and W. Shi, "Federated learning of predictive models from federated electronic health records," *J. of Medical Informatics*, vol. 112, pp. 57–69, 2018.

[22] S. Wolfert, L. C. Verdouw, and M. Bogaardt, "Big data in smart farming: A review," *Elsevier J. on Agricultural Systems*, vol. 153, pp. 60–80, 2017.

[23] Y. Jiber, H. Harroud, and A. Karmouch, "Precision agriculture monitoring framework based on wsn," in *2011 7th International Wireless Communications and Mobile Computing Conference*, 2011, pp. 2015–2020.

[24] V. Sharma, I. You, F. Palmieri, D. N. K. Jayakody, and J. Li, "Secure and energy-efficient handover in fog networks using blockchain-based dmm," *IEEE Communications Magazine*, vol. 56, no. 5, pp. 22–30, 2018.

[25] H. Zeng, S. R. Kulkarni, and H. V. Poor, "Attribute-distributed learning: Models, limits, and algorithms," *IEEE Tran. on Signal Processing*, vol. 59, pp. 386–398, 2011.

[26] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," *J. Communications of the ACM*, vol. 51, pp. 107–113, 2008.

[27] " Apache Hadoop," http://hadoop.apache.org/, 2018.

[28] M. Li, D. G. Andersen, J. W. Park, A. J. Smola, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, and B. Su, "Scaling distributed machine learning with the parameter server," in *USENIX Symposium on Operating Systems Design and Implementation*, 2014.

[29] M. Li, D. G. Andersen, J. W. Park, J. Smola, A. Ahmed, and V. Josifovski, "Scaling distributed machine learning with the parameter server," in *11th USENIX Symposium on Operating Systems Design and Implementation*, 2014.

[30] C. Xu, K. Wang, P. Li, S. Guo, J. Luo, B. Ye, and M. Guo, "Making big data open in edges: A resource-efficient blockchain-based approach," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 4, pp. 870–882, 2019.

[31] J. Konecny, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," in *IEEE Neural Information Processing Systems Workshop*, 2016.

[32] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," *arXiv:1602.05629*, vol. 2, 2016.

[33] G. Lan, S. Lee, and Y. Zhou, "Communication-Efficient Algorithms for Decentralized and Stochastic Optimization," https://arxiv.org/abs/1701.03961, 2017.

[34] R. Hasegawa and K. Hotta, "Plsnet: Hierarchical feature extraction using partial least square regression for image classification," *IEEE Tran. on Electrical and Electronic Engineering*, vol. 12, pp. 91–96, 2017.

[35] T. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng, and Y. Ma, "Pcanet: A simple deep learning baseline for image classification?" *IEEE Tran. on Image processing*, vol. 24, pp. 5017–5032, 2014.

[36] R. Hasegawa and K. Hotta, "Plsnet: A simple network using partial least square regression for image classification," in *IEEE Conf. on Pattern Recognition (ICPR)*, Cancun Center, Cancun, Mexico, 2016.

[37] M. Gowanlock, C. M. Rude, D. M. Blair, J. D. Li, and V. Pankratius, "A hybrid approach for optimizing parallel clustering throughput using the gpu," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 4, pp. 766–777, April 2019.

[38] K. Song, L. Li, S. Li, L. Tedesco, H. Duan, Z. Li, K. Shi, J. Du, Y. Zhao, and T. Shao, "Using partial least squares artificial neural network for inversion of inland water chlorophyll-a," *IEEE Tran. on Geoscience and Remote Sensing*, vol. 52, 2014.

[39] R. Hasegawa and K. Hotta, "Stacked partial least square regression for image classification," in *3rd Asian Conf. on Pattern Recognition*, 2015.

[40] V. D. Justafort, R. Beaubrun, and S. Pierre, "A hybrid approach for optimizing carbon footprint in intercloud environment," *IEEE Transactions on Services Computing*, vol. 12, no. 2, pp. 186–198, 2019.

[41] O. B. Akan, O. Cetinkaya, C. Koca, and M. Ozger, "Internet of hybrid energy harvesting things," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 736–746, 2018.

[42] D. C. Rose, W. J. Sutherland, C. Parker, M. Lobley, M. Winter, C. Morris, S. Twining, C. Ffoulkes, T. Amano, and L. V. Dicks, "Decision support tools for agriculture: Towards effective design and delivery," *Elsevier J. of Agricultural Systems*, vol. 149, pp. 165–174, 2016.

[43] N. Alexandratos and M. Bruinsma, "World agriculture towards 2030/2050, food and agriculture organization in the united nations," in *EAS working Paper No 12-03*, 2012.

[44] D. Vasisht, Z. Kapetanovic, J. Won, X. Jin, R. Chandra, A. Kapoor, S. N. Sinha, M. Sudarshan, and S. Stratman, "Farmbeats: An iot platform for data-driven agriculture," in *USENIX Symposium on Operating Systems Design and Implementation*, 2017.

[45] P. H. Garthwaite, "An interpretation of partial least squares," *American Statistical Association*, vol. 89, no. 425, pp. 122–127, 1994.
[46] H. Winning, "Standardization of ftir instruments using foss," *FOSS*, vol. 1, 2004.
[47] A. Nilsson, S. Smith, G. Ulm, E. Gustavsson, and M. Jirstrand, "A performance evaluation of federated learning algorithms," in *DIDL '18: Proceedings of the Second Workshop on Distributed Infrastructures for Deep Learning*, 2018.
[48] H. A. Neto, W. L. F. Tavares, D. C. S. Z. Ribeiro, R. C. O. Alves, L. M. Fonseca, and S. V. A. Campos, "On the utilization of deep and ensemble learning to detect milk adulteration," *BioData Mining*, 2019.
[49] Q. Wang, S. Guo, J. Liu, C. Pan, and L. Yang, "Profit maximization incentive mechanism for resource providers in mobile edge computing," *IEEE Transactions on Services Computing*, pp. 1–1, 2019.
[50] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, D. A. S. Reed, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015.
[51] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.
[52] J. Weng, J. Weng, M. Li, Y. Zhang, and W. Luo, "Deepchain: Auditable and privacy-preserving deep learning with blockchain-based incentive," *IACR Cryptology ePrint Archive*, vol. 2018, p. 679, 2018.
[53] G. Y. Wong, F. H. F. Leung, and S.-H. Ling, "A hybrid evolutionary pre-processing method for imbalance datasets," *Elsevier J. of Information Sciences*, vol. 454, pp. 161–177, 2018.
[54] H. Zheng, S. R. Kulkarni, and H. V. Poor, "Dimensionally distributed learning models and algorithm," in *IEEE Conf. on Information Fusion*, Cologne, Germany, 2008.
[55] C. Kulatunga, K. Bhargava, D. Vimalajeewa, and S. Ivanov, "Cooperative in-network computation in energy harvesting device clouds," *Sustainable Computing: Informatics and Systems*, vol. 16, pp. 106–116, 2017.
[56] K. Metwally, A. Jarray, and A. Karmouch, "A distributed auction-based framework for scalable iaas provisioning in geo-data centers," *IEEE Transactions on Cloud Computing*, pp. 1–1, 2018.
[57] M. M. Tajiki, M. Shojafar, B. Akbari, S. Salsano, M. Conti, and M. Singhal, "Joint failure recovery, fault prevention, and energy-efficient resource management for real-time sfc in fog-supported sdn," *Elsevier Computer Networks*, 2019.

Dixon Vimalajeewa received his B.Sc in mathematics and statistics from The University of Ruhuna, Sri Lanka in 2012, M.Sc in Computational Engineering from The Lappeenranta University of Technology, Finland in 2015 and PhD in Distributed data analytics from Waterford Institute of Technology in 2020.

Currently, he is a postdoctoral researcher at Telecommunications Software and Systems Group (TSSG) at Waterford Institute of Technology (WIT). His research interests include data analytics, sensor-based animal phenotypes and distributed learning algorithms (dvimalajeewa@tssg.org)

Chamil Kulatunga received his Bachelor in Electronics and Telecommunication Engineering from the University of Moratuwa (Sri Lanka) in 1999, Masters in Computer Science from the Waterford Institute of Technology (Ireland) in 2003, and PhD in Internet Engineering from the University of Aberdeen (UK) in 2009.

He was an Experienced Postdoctoral Researcher (2015-2017) under the Science Foundation Ireland funded PrecisonDairy project at the Telecommunications Software and Systems Group, Waterford Institute of Technology, Ireland. He is currently a Research Data Analyst under the Science Foundation Ireland and Origin Enterprises funded CONSUS project at the University College Dublin, Ireland.

His current research interests include distributed machine learning and data-driven agriculture in crop and dairy farming (chamilkul@gmail.com).

Donagh P Berry received his Bachelor in Agricultural Science and PhD in quantitative genetics at University College Dublin, Ireland in 2000 and 2003, respectively and a Masters in Bioinformatics and Systems Biology from University College Cork in 2012.

He is currently a senior principal investigator in quantitative geneticist at Teagasc, Ireland as well as being director of the VistaMilk Agri-Tech Research Centre. He holds professorships at three (inter)national universities. In his Teagasc capacity, he is responsible for the research on genetics in dairy cattle and is responsible for the development and implementation of genomic evaluations in dairy cattle, beef cattle and sheep in Ireland. As director of VistaMilk, he leads a team of $>$200 scientists in the development and deployment of digital technologies in precision dairy production (donagh.berry@teagasc.ie).

Sasitharan Balasubramaniam received the bachelor's degree in electrical and electronic engineering from The University of Queensland in 1998, the master's degree in computer and communication engineering from the Queensland University of Technology in 1999, and the PhD degree from The University of Queensland in 2005.

He is currently an Academy of Finland Research Fellow at the Department of Electronic and Communication Engineering, Tampere University of Technology, Finland, and an Acting Director of Research at the Telecommunications Software and Systems Group, Waterford Institute of Technology, Ireland, where he was involved in a number of Science Foundation Ireland projects. His current research interests include molecular and nanocommunications, and Internet of (bio–nano) Things. He is on the Steering Committee of the ACM NanoCom Conference which he co-founded. In 2018, he received the ACM/IEEE NanoCom Outstanding Milestone Award, and he is also the IEEE Nanotechnology Council Distinguished Lecturer. He is currently an Editor of the IEEE Internet of Things Journal, Nano Communication Networks (Elsevier), and Digital Communication Networks.