

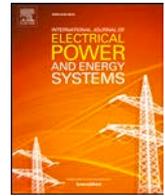
This document is published at:

Velasco, J.A.; Amaris, H.; Alonso, M. (2020). Deep Learning loss model for large-scale low voltage smart grids. *International Journal of Electrical Power & Energy Systems*, 121, 106054.

DOI: <https://doi.org/10.1016/j.ijepes.2020.106054>

This work is licensed under a **Creative Commons Attribution 4.0 International License**.





Deep Learning loss model for large-scale low voltage smart grids

Jose Angel Velasco, Hortensia Amaris*, Monica Alonso

Electrical Engineering Department, Universidad Carlos III de Madrid, Madrid, Spain

ARTICLE INFO

Keywords:

Deep learning
Smart grids
Uncertainty
Probabilistic modelling
Clustering
Phase imbalance

ABSTRACT

Distribution systems operators (DSOs) encounter the challenge of managing network losses in large geographical areas with hundreds of secondary substations and thousands of customers and with an ever-increasing presence of renewable energy sources. This situation complicates the estimation process of power loss, which is paramount to improve the network energy efficiency level in the context of the European Union energy policies. Thus, this article presents a methodology to estimate power losses in large-scale low voltage (LV) smart grids. The methodology is based on a deep-learning loss model to infer the network technical losses considering a large rollout of smart meters, a high penetration of distributed generation (DG) and unbalanced operation, among other network characteristics. The methodology has been validated in a large-scale LV distribution area in Madrid (Spain). The proposed methodology has proven to be a potential network loss estimation tool to improve the energy efficiency level in large-scale smart grids with a high penetration of distributed resources. The accuracy of the proposed methodology outperforms that of the state-of-the-art loss estimation methods, exhibiting a rapid convergence which allows for its use in real-time operations.

1. Introduction

Currently, one of the main challenges that distribution system operators (DSOs) encounter is improving the energy efficiency of smart grids; this entails knowing the degree of energy losses produced, especially in low voltage (LV) distribution networks. These energy losses include technical losses (TLs), produced by the Joule effect in cables, and non-technical losses (NTLs) which are related to electric fraud, meter tampering, and billing errors [1].

TLs are often computed through power flow techniques. However, this requires perfect knowledge of the grid topology, exact information regarding power consumption of each customer, and the power injection of each generation unit. The large deployment of smart meters in LV distribution networks, which are being installed in many countries, creates new possibilities for obtaining more precise information about the energy consumed by telemetered customers. Nevertheless, this information cannot be used directly when large distribution areas are considered. In these situations, the number of smart meter measurements may become unmanageable, and executing thousands of power-flows or state estimation algorithms in real-time is impractical. Additionally, the presence of intermittent distributed generation (DG), electric vehicles (EVs), or energy storage (ES) makes the evaluation of power losses more uncertain. Moreover, in some situations, uncertainty exists about customer network connectivity (phase or feeder), energy

demand (NTLs, non telemetered customers), or DG variability (due to weather condition dependency), thereby limiting the application of deterministic power flow techniques [2,3].

Several methods to estimate TLs can be found in literature where the use of load profiles and regression analysis have been proposed for loss estimation [4,5]. These are primarily applied to small-size distribution networks. Nonetheless, these techniques require accurate network data information (i.e., exact knowledge about the network topology and customer demand). Consequently, their applicability to large-scale distribution areas is limited because accurate network data are not always available.

Techniques based on the loss factor (LsF) methodology have been applied in distribution networks using reduced network data information [6,7]. LsF methods consist of computing a loss factor at representative feeders using, as inputs, the peak power demand, the customer load profile, and the feeder length. A disadvantage of these methods is that they ignore the existence of DG, whose presence is ever-increasing in LV distribution networks, primarily as rooftop PV units, and which has been demonstrated to have a relevant impact on network losses [8]. Moreover, these techniques do not provide real-time information because they are based on peak power demands.

Depending on the DG penetration level and the spatial-temporal correlation between load demand and DG injection, network losses can be either reduced or increased [9,10]. This fact aggravates the process

* Corresponding author.

E-mail address: hortensia.amaris@uc3m.es (H. Amaris).

of network loss estimation using only, as input, the data recorded by customers' smart meters and by the secondary substation (SS) energy meters. Therefore, any loss estimation method for smart grids must consider the presence of DG to provide an accurate loss estimation.

Top-down and bottom-up approaches have been proposed in [11] to address the lack of accurate network data in large areas. In the first step (top-down), the different areas are clustered according to some explanatory variables, and representative feeders are defined for each cluster group. In the second stage, loss functions are obtained through power flow equations for each representative feeder cluster. Losses for each feeder are obtained according to the load that each feeder carries. Notwithstanding, the method has been applied only in balanced medium voltage (MV) distribution networks. Consequently, the application of this methodology to unbalanced LV smart grids would provide incorrect results.

The concept of representative feeders within a large distribution area appears to be a potential approach to infer power loss levels in LV networks [12,13]. The accuracy of the loss estimation will depend on the similarity between the representative feeders (i.e., a feeder whose properties can be extrapolated to all the feeders of the same cluster) and the remainder of feeders belonging to that cluster [14]. A recent overview of representative feeder works can be found in [15,16].

As distribution feeders have different topological structures and characteristics (e.g., number of clients, line length, energy supply), some authors have proposed the application of a clustering process to obtain losses in a set of representative feeders [17,18]. In [17], a Fuzzy C-Number (FCN) clustering technique is used to obtain a loss equation by applying a cluster-wise fuzzy regression analysis. In [18], a clustering-based method is presented to locate feeders that have similar characteristics. For each cluster obtained, the main feeder is selected to estimate the network's technical losses. The application to large-scale areas is achieved using an iterative process in which random feeder parameters are selected to create new network topologies. Although the method has been designed for LV distribution areas, it does not take into account the unbalanced operation of these networks [19]. Consequently, the loss estimation that this method provides is useful only for balanced feeders.

Recently, machine learning techniques have also been applied to the estimation of losses in distribution networks [20–24]. In [21], eXtreme Gradient Boosting (XGBoost) is used to estimate statistical technical losses of distribution feeders. The method requires both a high number of representative feeders and a large amount of historical data to apply the clustering feeder stage. This is an important limitation for expanding its applicability to large-scale smart grids characterised by reduced network data availability. In [25], a feedforward Artificial Neural Network (ANN) for feeder loss analysis is proposed. Key factors of feeder losses, such as feeder loading, power factor, feeder length, and transformer capacity, are used as input variables to the network. The proposed method provides accurate results but does not address the scalability to large-scale problems. Based on the same idea, [22] detailed an ANN-based model to estimate power losses in distribution and transmission networks where DG has not been considered.

This article presents an innovative formulation for the estimation of power losses in large geographical LV smart grids characterised by high DG penetration and unbalanced operation. The method is built upon an enhanced representative feeder selection process and a novel deep-learning loss model, and it considers the network TUs in an entire geographical area. This information will assist the DSO in determining the variability of technical losses in large distribution areas using only the smart meter measurements of demand customers and DG generation measurements.

The principal contributions and novelties of the presented work can be summarised as follows:

- A novel deep learning methodology for loss estimation in large-scale LV smart grids with a high penetration of DG.

- The proposed model considers the unbalanced operation of LV networks. This aspect is not often considered in other literature.
- The proposed method can cope with uncertainty in the network, energy consumption, and distribution generation production.

To the best of the authors' knowledge, the estimation of power losses in unbalanced large-scale LV smart grids using deep learning techniques remains highly unexplored in the literature.

2. Methodology

Actual LV distribution networks are characterised by a massive roll-out of smart metering deployment among LV residential customers [26]. Consequently, active network losses at a certain time instant could be expected to be easily calculated by summing all the smart meter measurements of active imported power (consumption) and subtracting them from the injected active power measured at the secondary substation energy meters.

Nevertheless, this intuitive scheme is only valid under the following situations:

- All the customers are telemetered.
- There is no presence of DG.
- There is no presence of electric fraud (no illegal connections and no manipulated metering devices).
- There are no loops, i.e., the configuration of the network is radial.

In real LV distribution networks, the majority of residential and small-size commercial customers are telemetered, but some commercial and industrial customers are still not telemetered because they are required to provide energy consumption on only a monthly basis [27]. Thus, the simple assumption of summing all demand smart meter measurements and subtracting them from the injected power measured at the secondary substation meter is clearly unacceptable for loss estimation in LV networks if there are non-telemetered customers.

This section provides the formulation of the proposed technical estimation methodology for power losses in large-scale LV smart grids which could easily be composed of hundreds of SS and thousands of distribution feeders. Each feeder belonging to that area can have different topological characteristics (such as cross-section, length, and overhead/underground configuration) and different levels of DG and smart meter penetration, voltage unbalance, and load demand. The methodology comprises the following five steps:

1. **Data collection.**
2. **Data normalisation.**
3. **Features extraction.**
4. **Feeder clustering.**
5. **Deep Neural Network technical losses model.**

The whole process is shown in Fig. 1.

2.1. Data collection

Large LV distribution areas are characterised by high levels of variability such as feeder properties, number and spatial location of customers, and number and spatial location of DG units. Therefore, the network input dataset to be applied to the deep learning model has to be sufficiently representative to describe the nature and behaviour of the feeders, but it also has to be reduced to prevent overfitting the deep learning model. The more relevant characteristics of feeders are summarised in Table 1, which include information regarding the level of unbalance, the presence of DG units, the ratio of customers with smart meters, and information regarding the physical configuration of the feeders (length, section, configuration, etc.). Note that in total, there are fourteen customised feeder characteristics, because characteristics with

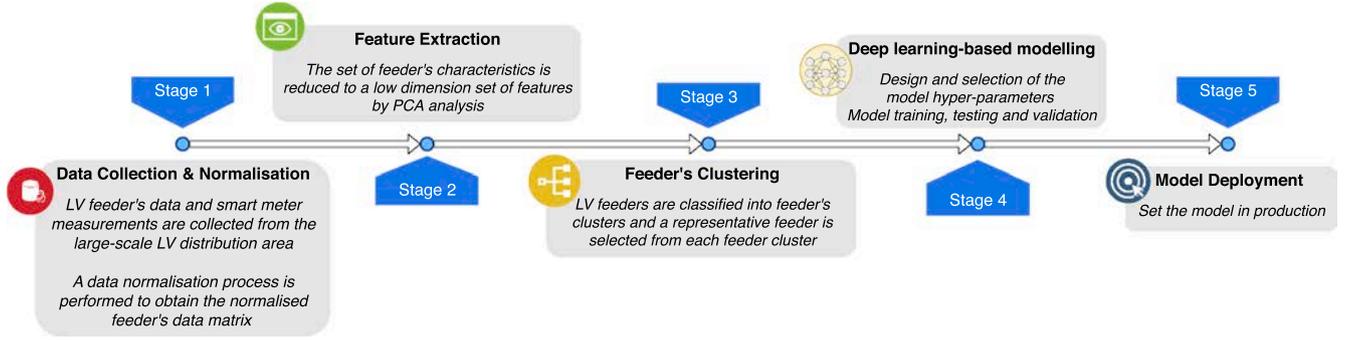


Fig. 1. DNN loss methodology.

sub-index p are defined for each phase $p \in \{a, b, c\}$.

As the majority of LV feeders have radial topology and the presence of loops (weakly meshed topologies) is scarce, the location of the DG units along the feeder is crucial to determine the levels of power loss [10]. Hence, the spatial location of the DG units (X_s^f) is determined based on the Euclidean distance (d_e^*) between the centroid of the DG unit coordinates and the SS coordinates. The characteristic X_s^f will have values according to Eq. (1). Depending on the value of that characteristic, three possible situations exist: DG located at the main feeder head, feeder centre, or end of the feeder, as illustrated in Fig. 2.

$$X_s^f = \frac{d_e^*}{L} - \frac{1}{2} = \begin{cases} -\frac{1}{2} & \text{if } d_e^* = 0 \rightarrow \text{Feeder Head} \\ 0 & \text{if } d_e^* = \frac{L}{2} \rightarrow \text{Feeder Centre} \\ \frac{1}{2} & \text{if } d_e^* = L \rightarrow \text{Feeder Tail} \end{cases} \quad (1)$$

where L denotes the length of the feeder and d_e^* is the Euclidean distance obtained using (2):

$$d_e^* = \sqrt{(X_{ss} - X^*)^2 + (Y_{ss} - Y^*)^2} \quad (2)$$

where (X_{ss}, Y_{ss}) and (X^*, Y^*) are the GPS coordinates of the SS and the DG unit, respectively calculated using (3):

$$X^* = \frac{\sum_{i=1}^{N_{DG,f}} X_i}{N_{DG,f}}, Y^* = \frac{\sum_{i=1}^{N_{DG,f}} Y_i}{N_{DG,f}} \quad (3)$$

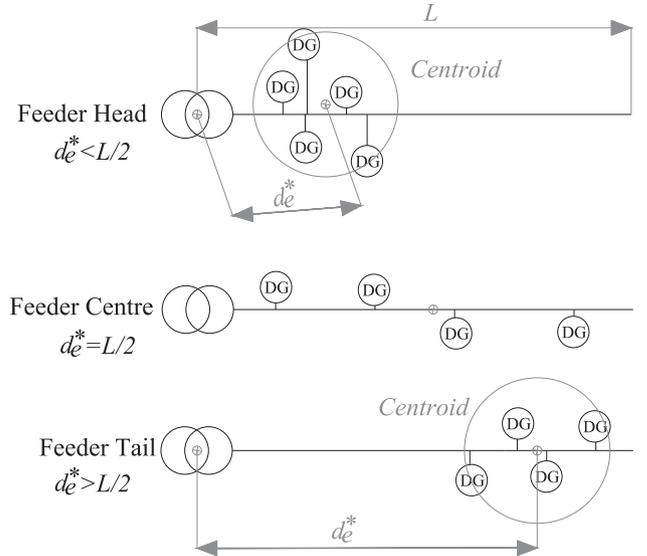


Fig. 2. Characterisation of the spatial location of the DG units along the feeder.

(X_i, Y_i) are the coordinates of the i^{th} DG unit belonging to feeder f , and $N_{DG,f}$ is the number of DG units connected to feeder f .

The collection of feeder characteristics can be described in

Table 1

Properties of feeders.

Property	Characteristic	Description	Data source
$X_{1,p}^f$	Unbalance power level phase p	Power ratio between power contracted by customers connected to phase $p \in \{a, b, c\}$ of feeder f and power contracted by all the customers connected to the same feeder	SM
X_2^f	Feeder loading level	Ratio between power contracted by the customers connected to the feeder f and power rating of the transformer located in the SS from which the feeder belongs	N
X_3^f	Smart meters deployment	Ratio between the number of telemetered customers and the total number of customers connected to the feeder	N
X_4^f	DG penetration level	Ratio between the DG power and the power demand of customers connected to the feeder	N
X_5^f	DG spatial location	Spatial-location of the DG units along the feeder using the projected Euclidean distance respect to the location of the SS	N
X_6^f	Customers spatial location	Spatial-location of the customers along the feeder using the projected Euclidean distance respect to the location of the SS	N
X_7^f	Self-consumption ratio	Ratio between the average energy produced by the DG unit of a customer and its average energy consumption	N
X_8^f	Impedance feeder path	Total feeder impedance	N
$X_{9,p}^f$	Demand MV WD phase p	Mean Value (MV) of the power demand of phase $p \in \{a, b, c\}$ of the feeder f in Working days (WD)	SM
$X_{10,p}^f$	Demand SD WD phase p	Standard Deviation (SD) of the power demand of phase $p \in \{a, b, c\}$ of the feeder f in WD	SM
$X_{11,p}^f$	Demand MV NWD phase p	Mean Value (MV) of the power demand of phase $p \in \{a, b, c\}$ of the feeder f in Non-Working days (NWD)	SM
$X_{12,p}^f$	Demand SD NWD phase p	Standard Deviation (SD) of the power demand of phase $p \in \{a, b, c\}$ of the feeder f in NWD	SM
X_{13}^f	Ratio customers per feeder	Ratio between number of customers connected to a feeder f and the total number of customers connected to the same SS	N
$X_{14,p}^f$	Ratio customers per phase	Ratio between the customers connected to the phase $p \in \{a, b, c\}$ of feeder f and the total number of customers connected to the same feeder	N

MV: Mean Value, SD: Standard Deviation, WD: Working days, NWD: Non-Working days, SM: Smart Meters Data, N: Network Data. p is the index for phases: $\{a, b, c\}$.

mathematical terms as a feeder data matrix $X \in \mathbb{R}^{n \times \tilde{p}}$ where the columns are the feeder characteristics and the rows are the feeder samples as indicated in (4).

$$X = \begin{pmatrix} X_{1,1} & X_{1,2} & \dots & X_{1,\tilde{p}} \\ X_{2,1} & X_{2,2} & \dots & X_{2,\tilde{p}} \\ \vdots & \vdots & \ddots & \vdots \\ X_{n,1} & X_{n,2} & \dots & X_{n,\tilde{p}} \end{pmatrix} \quad (4)$$

2.2. Data normalisation

A normalisation process is performed to address the diverse nature of the properties to obtain a more interpretable description of the feeder characteristics collected and to provide the same weight to all feeder properties [28]. For instance, the typical length of LV feeders may vary between 50 and 500 m, while the typical phase unbalance may vary between 0.25 and 0.35. If there is no data normalisation stage, the phase unbalance variation will be negligible compared with the feeder length.

The original feeder data matrix $X \in \mathbb{R}^{n \times \tilde{p}}$ is transformed into the normalised feeder data matrix $M \in \mathbb{R}^{n \times \tilde{p}}$ by applying to each element $X_{i,j}$ of matrix X a normalising function $f: \mathbb{R} \rightarrow \mathbb{R}$ defined in Eq. (5) [29].

$$x_{i,j} = \frac{X_{i,j} - \min\{X_j\}}{\max\{X_j\} - \min\{X_j\}}, \quad \forall i \in (1, \dots, n), \quad \forall j \in (1, \dots, \tilde{p}) \quad (5)$$

where

- $x_{i,j}$ is the normalised element $X_{i,j}$ of matrix X ,
- $\min\{X_j\}$ is the minimum value of the characteristic j , (i.e. the minimum value of the column j of matrix X),
- $\max\{X_j\}$ is the maximum value of the characteristic j .

The resulting normalised feeder data matrix M is indicated in (6).

$$M = \begin{pmatrix} X_{1,1} & X_{1,i} & X_{1,\tilde{p}} \\ X_{2,1} & X_{2,i} & X_{2,\tilde{p}} \\ \vdots & \dots & \vdots \\ X_{n,1} & X_{n,i} & X_{n,\tilde{p}} \end{pmatrix} \quad (6)$$

Note that each column of matrix M corresponds with a normalised feeder characteristic M_j (7) of dimensions $(1 \times n)$, that is, the M_j vector contains the j^{th} characteristic of the n feeders of the LV smart grid.

$$M_j = (x_{1,j}, \dots, x_{n,j})^T, \quad \forall j \in (1, \dots, \tilde{p}) \quad (7)$$

2.3. Features extraction

Large-scale LV smart grids can comprise hundreds of substations and thousands of feeders; consequently, managing such a large dataset could lead to a high computational burden and overfitting problems. In such situations, reducing the dimensions of the problem and simultaneously retaining the maximum original data variability becomes necessary. One of the most well-known dimension reduction techniques is the principal component analysis (PCA) [30], which has been used in different applications such as finance, biology, etc. [31].

PCA is an unsupervised learning method that reduces the dimensionality of a correlated set of variables into a set of linearly independent uncorrelated variables (which are known as principal components, PC), maintaining the majority of the variability existing in the initial dataset [31]. This transformation is conducted in the sense that the first component has the largest variance (e.g., it captures the maximum possible variability of the initial dataset) and the second component has the highest variance subject to the constraint that it is orthogonal to the first component.

The objective is to determine a linear mapping given by the projection matrix $\tilde{U} \in \mathbb{R}^{\tilde{p} \times s}$ which transforms each row of matrix M (i.e., each normalised feeder observation) expressed as the vector $M_i = [x_{i,1}, \dots, x_{i,\tilde{p}}] \in \mathbb{R}^{1 \times \tilde{p}}, \forall i \in (1, \dots, n)$, into a lower dimension representation, $\tilde{M} \in \mathbb{R}^{\tilde{p} \times s}$ (8) with dimension $s < \tilde{p}$. Each column of matrix \tilde{M} is a PC.

$$\tilde{M} = \tilde{U}^T M^T \quad (8)$$

The number of PCs is often selected by defining the total variability to be captured. The number of PCs selected to capture at least a percentage (η) of the cumulative variability has to fulfil (9) where λ_{kp} is the kp^{th} eigenvalue associated and β is the total number of eigenvalues of the covariance matrix S

$$\operatorname{argmin}_s \frac{\sum_{kp=1}^s \lambda_{kp}}{\sum_{kp=1}^{\beta} \lambda_{kp}} > \eta(\%) \quad (9)$$

The covariance matrix $S \in \mathbb{R}^{\tilde{p} \times \tilde{p}}$ (10) can be obtained by computing, in the diagonal terms, the variances s_j^2 of each variable (column) of matrix M and, in the off-diagonal elements, the covariances $s_{j,j'}, \forall j, j' \in (1, \dots, \tilde{p}), j' \neq j$, among each pair of variables (columns) of matrix M .

$$S = \begin{pmatrix} s_1^2 & s_{1,2} & \dots & s_{1,\tilde{p}} \\ s_{2,1} & s_2^2 & \dots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ s_{\tilde{p},1} & \dots & \dots & s_{\tilde{p}}^2 \end{pmatrix} \quad (10)$$

2.3.1. First PC ($kp = 1$)

The first projection vector \tilde{U}_1 has to be obtained by solving the optimisation problem defined in (11) which includes the maximisation of the quantity $\tilde{U}_1^T S \tilde{U}_1$ and the normalisation constraint $\tilde{U}_1^T \tilde{U}_1 = 1$.

$$\operatorname{argmax}_{\tilde{U}_1} \tilde{U}_1^T S \tilde{U}_1 \text{ subject to } \tilde{U}_1^T \tilde{U}_1 = 1 \quad (11)$$

This can be solved by using the technique of the Lagrangian multipliers, from which the Eq. (11) is converted in (12) where λ_1 is the Lagrangian multiplier.

$$\psi_1 = \tilde{U}_1^T S \tilde{U}_1 - \lambda_1 (\tilde{U}_1^T \tilde{U}_1 - 1) \quad (12)$$

Eq. (12) can be differentiated with respect to \tilde{U}_1 , resulting in (13).

$$\frac{\partial \psi_1}{\partial \tilde{U}_1} = 0 \Rightarrow (S - \lambda_1 I) \tilde{U}_1 = 0 \quad (13)$$

where $I \in \mathbb{R}^{\tilde{p} \times \tilde{p}}$ is the identity matrix. By the definition of eigenvalues and eigenvectors, from (13), λ_1 is clearly the eigenvalue of the covariance matrix S , and \tilde{U}_1 is the corresponding eigenvector. The result obtained in (13) indicates that the first projection vector \tilde{U}_1 will be the eigenvector associated with the first large eigenvalue λ_1 of the covariance matrix S .

2.3.2. Second PC ($kp = 2$)

Obtaining the second PC (i.e., the second column of the reduced feeder data matrix \tilde{M}) requires the eigenvector \tilde{U}_k with $k = 2$ to be determined, which also implies solving the optimisation problem (11) but also includes the constraint that the projection by \tilde{U}_2 is uncorrelated with \tilde{U}_1 , as indicated in (14).

$$\operatorname{argmax}_{\tilde{U}_1} \tilde{U}_1^T S \tilde{U}_1 \text{ subject to } \tilde{U}_1^T \tilde{U}_1 = 1, \quad \tilde{U}_2^T \tilde{U}_1 = 0 \quad (14)$$

Subsequently, the procedure of Lagrangian multipliers is followed to obtain \tilde{U}_2 , but in this case, there are two constraints to consider; thus, two multipliers must be defined as indicated in (15).

$$\psi_2 = \tilde{U}_2^T S \tilde{U}_2 - \lambda_2 (\tilde{U}_2^T \tilde{U}_2 - 1) - \mu_2 (\tilde{U}_2^T \tilde{U}_1) \quad (15)$$

Differentiating (15) with respect to \tilde{U}_2 results in (16).

$$\frac{\partial \psi_2}{\partial \tilde{U}_2} = 0 \Rightarrow S \tilde{U}_2 - \lambda_2 \tilde{U}_2 - \mu_2 \tilde{U}_1 = 0 \quad (16)$$

Multiplying both sides of Eq. (16) by \tilde{U}_1^T results in (17).

$$\tilde{U}_1^T S \tilde{U}_2 - \lambda_2 (\tilde{U}_1^T \tilde{U}_2) - \mu_2 (\tilde{U}_1^T \tilde{U}_1) = 0 \rightarrow \mu_2 = 0 \quad (17)$$

The first and second terms of Eq. (17) are zero due to the zero-correlation constraint. The third term is the unity due to the normalisation constraint. Therefore, this results in $\mu_2 = 0$, which enables us to rewrite the Eq. (16) into a reduced form (Eq. (18)), where λ_2 is concluded as the second projection vector and \tilde{U}_2 is the eigenvector associated with the second large eigenvalue of the covariance matrix S .

$$\frac{\partial \psi_2}{\partial \tilde{U}_2} = 0 \Rightarrow (S - \lambda_2 I) \tilde{U}_2 = 0 \quad (18)$$

2.3.3. Subsequent PCs ($k_p > 2$)

As a result, the subsequent projection vectors \tilde{U}_{k_p} ($k_p = 3, \dots, s$) could be obtained as each eigenvector associated with the k_p^{th} large eigenvalue λ_{k_p} of the covariance matrix S .

2.4. Feeder clustering

In this stage, the n feeders from the target large-scale LV smart grids are partitioned into a reduced number of feeder groups (i.e., clusters), where feeders belonging to the same cluster are similar [32]. Subsequently, from each feeder cluster, a representative feeder is selected as the closest to the centroid of that cluster.

Table 2 summarises a comparison between the most extended algorithms of each clustering technique where \hat{k} is the number of clusters, s is the dimension of the reduced observation vector (columns of matrix \tilde{M}), n is the number of samples (rows of matrix \tilde{M}), and ϵ is the number of iterations of the Expectation–Maximisation (EM) algorithm.

According to [13] the Gaussian mixture model (GMM) and improved K-means++ are suitable candidates for clustering feeders with many feeder characteristics. Nevertheless, GMM is characterised by a higher computational complexity in comparison with K-means++. Therefore, in this paper, the n feeders gathered in the reduced feeder data matrix \tilde{M} are clustered using the K-means++ algorithm, using the uncorrelated s PCs obtained in the PCA analysis 2.3.

2.4.1. Clustering algorithm

The K-means++ algorithm partitions the dimension-reduced feeder data matrix $\tilde{M} \in \mathbb{R}^{n \times s}$ of feeder characteristics (which have observations of n feeders and s PCs) into \hat{k} separated subsets (i.e., feeder clusters) $C_r = \{c_1, \dots, c_{n_r}\}$, $\forall r \in (1, \dots, \hat{k})$. K-means++ has two steps: the initialization of the centroids and the assignment step.

1. Centroid Initialisation: The location of the \hat{k} centroids are selected as follows:

(a.1) Random selection of the r^{th} cluster's centroid, $c_r^{*(0)} = c_i$ where $c_i \in \tilde{M}$ is selected from the feeder data points that compound the feeder PCs.

(a.2) Computation of the Euclidean distance \mathcal{D} between each feeder data point and the centroid $\mathcal{D}(c_i, c_r^{*(0)}) = \|c_i - c_r^{*(0)}\|^2$.

(a.3) Updating of the centroid $c_{r+1}^{*(0)}$ as the one with the highest probability according to (19).

$$\mathcal{P}(c_{r+1}^{*(0)} = c_j) = \mathcal{D}(c_j, c_r^{*(0)})^2 / \sum_{k, j \neq k} \mathcal{D}(c_k, c_r^{*(0)}) \quad (19)$$

(a.4) Repeat (a.2) and (a.3) until \hat{k} centroids have been selected.

2. Assignment step: After the initialisation of the centroids, each feeder data point is assigned to the closest centroid cluster by minimising the mean square error (MSE) according to (20).

$$MSE(C_1, \dots, C_{\hat{k}}) = \sum_{r=1}^{\hat{k}} \frac{1}{n_r} \sum_{i=1}^{n_r} \left\| c_i - c_r^{*(0)} \right\|^2, \quad r \in (1, \dots, \hat{k}) \quad (20)$$

Subsequently, the initial cluster set is updated by recalculating the centroids as the mean of the samples of the clusters using (21).

$$c_r^{*(0)} = \frac{1}{n_r} \sum_{i=1}^{n_r} c_i, \quad \forall r \in \left(1, \dots, \hat{k} \right) \quad (21)$$

The Assignment step (20)–(21) is repeated q times until the change in all $c_r^{*(q)}$ is sufficiently small ϵ as indicated in (22).

$$c_r^{*(q)} - c_r^{*(q-1)} \leq \epsilon, \quad \forall r \in (1, \dots, \hat{k}) \quad (22)$$

2.4.2. Clustering evaluation

Two indices can be used to evaluate whether a feeder sample has been appropriately associated with the right cluster: the Silhouette and the Global Silhouette (GS) indices. The Silhouette index is a coefficient that quantifies the similarity of the object (in this case, a feeder sample) with the remainder of the elements of the group belonging to its own cluster (in this case, the feeder group) [33]. The values that can adopt the Silhouette range from -1 (low relation) to $+1$ (high relation).

The Silhouette of the feeder sample c_i is defined as $\mathcal{S}(c_i)$ and is calculated using (23).

$$\mathcal{S}(c_i) = \frac{b(c_i) - a(c_i)}{\max\{a(c_i), b(c_i)\}} \approx 1 - \frac{a(c_i)}{\underbrace{b(c_i)}_{b(c_i) > a(c_i)}} \quad \forall i \in (1, \dots, n) \quad (23)$$

where

- $a(c_i)$ is the average distance between the feeder sample c_i and all the other feeders belonging to the same cluster.
- $b(c_i)$ is the smallest average distance between feeder sample c_i and all the others feeders in all clusters. Notice that for $b(c_i) > a(c_i)$ Eq. (23) can be reduced.

Table 2
Clustering techniques and performance comparison.

Technique	Algorithm	Time Complexity	Scalability	Large datasets	Outliers sensitivity	Noise sensitivity
Hierarchical	BIRCH	$O(n)$	High	Yes	Low	Low
Hierarchical	CURE	$O(\tilde{n}^2 \log(\tilde{n}))$	High	Yes	Low	Low
Hierarchical	ROCK	$O(n^2)$	Intermediate	No	Low	Low
Partition-based	K-means++	$O(\log(\hat{k}))$	Intermediate	Yes	High	High
Partition-based	K-medoids++	$O(\hat{k}(n - \hat{k})^2)$	Low	No	Low	Low
Partition-based	CLARANS	$O(n^2)$	Intermediate	Yes	Low	Low
Distribution-based	GMM	$O(\epsilon s^3)$	Intermediate	Yes	–	–

Feeder sample c_i would have a Silhouette index value $S(c_i) \sim 1$, if $b(c_i) \gg a(c_i)$, which implies that the feeder sample is poorly related with it neighbouring clusters, so the clustering configuration is appropriated. On the contrary, if the Silhouette index value is $S(c_i) \sim -1$, then the feeder sample is likely to belong to a neighbouring feeder cluster; thus the clustering requires revision. Finally, $S(c_i) \sim 0$ indicates that the feeder sample c_i is on the border of two neighbouring clusters.

The Global Silhouette index is used to evaluate the quality of the feeder clustering process (24). The GS index is based on the Silhouette index and provides a general sense of the quality of the clustering process. The higher the GS value, the higher the quality of the clustering.

$$GS = \frac{1}{\hat{k}} \sum_{r=1}^{\hat{k}} \frac{1}{n_r} \sum_{i=1}^{n_r} S(c_i) \quad (24)$$

where n_r is the number of feeders belonging to the cluster C_r , $\forall r \in (1, \dots, \hat{k})$.

2.4.3. Representative feeder selection

When the appropriate clusters have been obtained, the representative feeders, denoted by \tilde{c}_r are selected as the feeder sample closest to the centroid within each feeder cluster via the minimum Euclidean distance to the centroid c_r^* , which is described in (25).

$$\tilde{c}_r = \underset{c_i \in C_r}{\operatorname{argmin}} \{ \|c_i - c_r^*\|^2, \forall r \in (1, \dots, \hat{k}) \} \quad (25)$$

Note that the representative feeder \tilde{c}_r , selected from the cluster C_r , evaluates the power losses from the set of feeders belonging to that cluster.

In this paper, an average Euclidean distance factor α_r is used as a weighting factor to extrapolate the representative feeder to the remainder of the feeders inside the cluster C_r to address the differences between the representative feeder \tilde{c}_r and the different feeders belonging to that cluster $c_i \in C_r$ (as is illustrated in Fig. 3). This factor is based on the Euclidean distance from each feeder to the reference feeder, in the cluster, according to (26).

$$\alpha_r = \frac{1}{n_r} \sum_{i=1}^{n_r} \left\| c_i - \tilde{c}_r \right\|^2, r \in (1, \dots, \hat{k}) \quad (26)$$

3. Deep neural network loss model

Recently, a new group of machine learning models has been applied in the analysis of power systems [20,34,35]. The application of these techniques has emerged due to the increasing amount of data available and the increasing number of requirements related to real-time operation. Moreover, these models can reveal hidden insights in the data generated in the power network, which is something that traditional approaches cannot provide.

Power loss estimation can be classified as a regression task; consequently, the machine learning models suited for that purpose are Decision Trees and Neural Networks [36]. In the first group, the most widely used models are Random Forest and Gradient Boosting models, which produce accurate results but are slow for real-time operation

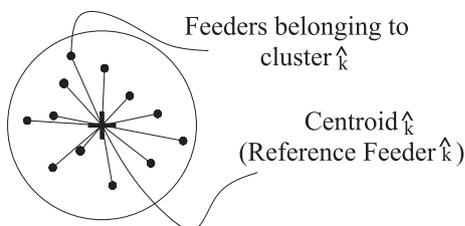


Fig. 3. Feeder cluster representation.

[37]. In the second group, Deep Neural Networks (DNNs) capture complex non-linear relationships (such as power loss estimation) and represent a compromise between accuracy and performance. DNNs are computing systems inspired by the biological neural networks that constitute human brains and are classified as supervised machine learning. Without any prior knowledge, they automatically identify the input to provide a specific output depending on the training received [38].

DNNs can be Convolution Neural Networks (CNNs) if the data flow from the input layer to the output layer and backward again, or Feedforward Neural Networks (FNNs) where the data flows in only one direction from the input layer to the output layer. FNNs are multi-purpose, offering high performance and accuracy; hence, they are selected for power loss estimation.

3.1. DNN architecture

The DNN-based power loss model proposed in this paper is illustrated in Fig. 4. The model can be described physically as a collection of nodes called artificial neurones, which are the basic units of computation. These neurones are interconnected by weighted links called edges. The input layer L_{in} receives the data input vector denoted by \mathbb{X} , which comprises the representative demand and generation conditions of the whole LV distribution area, through the scaled demand and generation patterns of the representative feeders \tilde{c}_r . The output layer L_o provides the model output, \mathbb{Y} , which represents the total technical losses of the large-scale LV distribution area. Between the input and output layers are multiple hidden layers, $k \in (1, \dots, h)$, with h being the number of hidden layers, each one containing n_k number of neurones.

Moreover, to avoid overfitting (when the model performs so closely to the training data but fails with new data), some of the neurones of the model must be cancelled (i.e.: no output). This is achieved using the dropout technique.

3.2. Selection of the DNN hyper-parameters

A critical part of the modelling of the DNN is the selection of the DNN control parameters to characterise the model structure and performance. These control parameters are known as hyper-parameters and their configuration is crucial to the performance of the deep learning model. In deep learning models, determining the best configuration for hyper-parameters is not trivial [39]. In this paper, the grid search strategy is adopted to tune the hyper-parameters of the deep learning model because it is the method that provides the optimal combination of hyper-parameters [39].

The following hyper-parameters are defined for the deep learning power losses model:

- **Number of hidden layers, h :** Large Deep Neural Networks are likely to obtain better results because the model has more opportunities to learn independent representations.
- **Number of neurones per hidden layer, n_k :** A large number of neurones per hidden layer increase accuracy but, if the number is too high, it may cause the model to overfit. By contrast, a few neurones per hidden layer may cause the model to under-fit.
- **Dropout, ζ :** This is a regularisation technique to avoid overfitting which consists of cancelling some neurones of the hidden layers [40]. A widely-used good starting point is the cancellation of 20% of the hidden neurones. Cancelling a low number of neurones produces a minimal effect, while cancelling a higher number of neurones may cause the model to under-train.
- **Learning rate, η :** This a coefficient to control how the model changes depending on the error response, as the weights of the deep learning model are updated based on that error.
- **Weights initialisation, $\{w_{i,j}(0), w_{i,j}(0)\}$:** The weight initialisation scheme depends on the activation function selected. In this paper, a

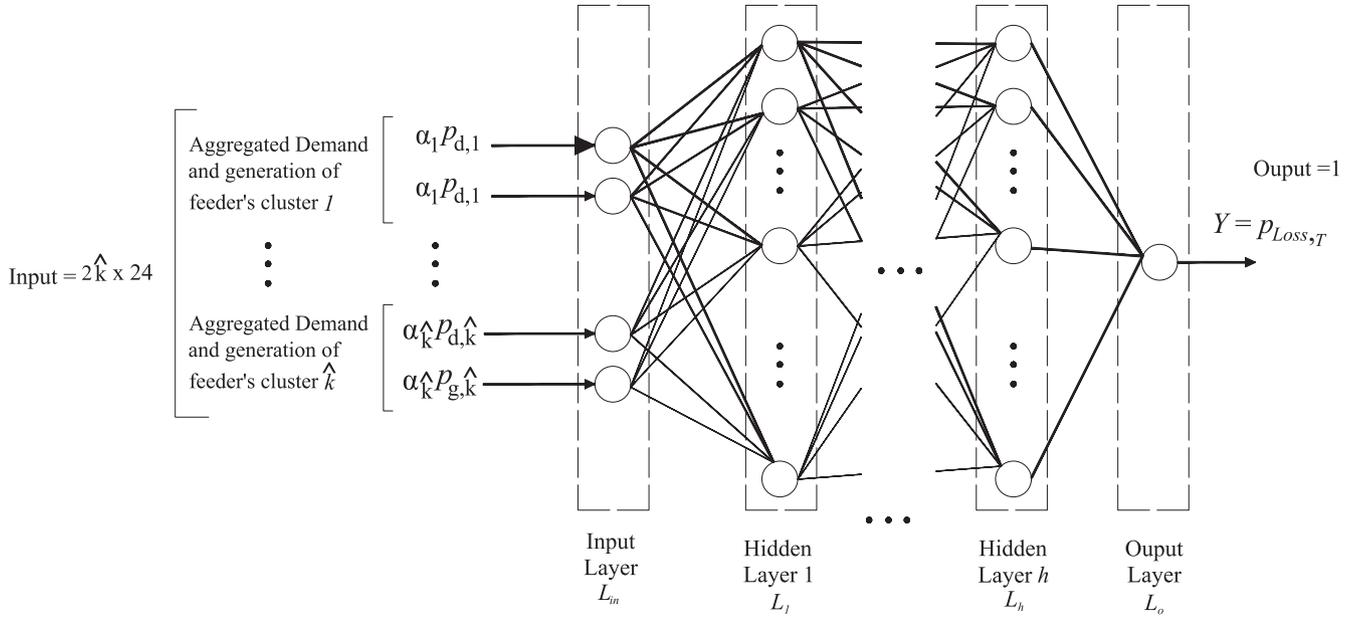


Fig. 4. DNN power loss model.

sigmoid function is selected; hence, in general, $w_{ij} \in (0, 1)$. The lower bound of the minimum weight can be defined randomly as $w_{ij}^{min}(0) \in (0, 0.5)$. Meanwhile, the upper bound of the maximum weight can be defined randomly as $w_{ij}^{max}(0) \in (0.5, 1)$.

- **Number of epochs, ϑ :** This is the number of times that the complete training dataset is offered to the model in the training stage. The number of epochs must be increased until the validation accuracy decreases.
- **Batch size, ν :** This is the number of training samples used as inputs to the model before the edge weights are updated. The training dataset can be divided into one or more batches. If the dataset is not divided evenly, some samples could be ignored.

3.3. DNN data input

As the entire set of feeders belonging to the large-scale LV distribution area has been separated into feeder clusters, the daily representative patterns (RPs) for the power demand and generation (DG) of each feeder cluster must be defined and used as input to the deep learning loss model. Note that power generation from the DG is intermittent and highly stochastic, imposing uncertainties in the power generation in each feeder clusters. Moreover, the load variability and existence of non-telemetered customers add more uncertainty to the power demand. Consequently, because of the existing uncertainties in both the power generation and load, variability is modelled in this paper with consideration to probabilistic techniques where the statistical properties of the uncertain variables are modelled using probability distribution functions (PDFs) based on the available historical data using Monte Carlo simulations and Kernel Density Estimation techniques [41].

Fig. 5 illustrates the process of scaling each daily aggregated pattern (AP) of each representative feeder to obtain the daily RPs of each feeder cluster. Therefore, the model input vector, \mathbb{X} , defined in (27), represents the representative demand and generation conditions of the entire LV distribution area at an instant of time, t . Specifically, \mathbb{X} contains the daily RPs of the demand \mathbb{X}_d and generation \mathbb{X}_g for each feeder clusters in which the entire set of feeders has been separated.

$$\mathbb{X} = [\mathbb{X}_{d,1} \quad \mathbb{X}_{g,1} \dots \mathbb{X}_{d,\hat{k}} \quad \mathbb{X}_{g,\hat{k}}]^T \quad (27)$$

Each demand and generation RP is obtained from the daily aggregated pattern of the representative feeder, denoted as $p_{d,r}(t)$ (28) for

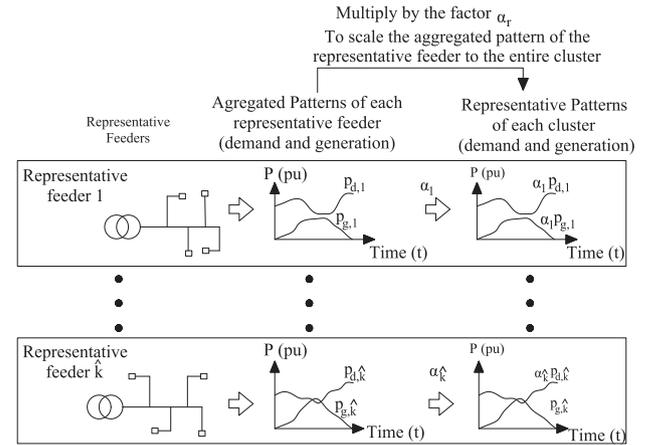


Fig. 5. Schematic process for obtaining the daily representative patterns.

demand and $p_{g,r}(t)$ (29) for generation, scaled to the entire feeder cluster using the extrapolation factor α_r , defined previously in (26).

$$\mathbb{X}_{d,r} = \alpha_r p_{d,r}(t), \quad \forall r \in (1, \dots, \hat{k}) \quad (28)$$

$$\mathbb{X}_{g,r} = \alpha_r p_{g,r}(t), \quad \forall r \in (1, \dots, \hat{k}) \quad (29)$$

Note that the existing uncertainty in both demand and generation is considered in this paper through the daily aggregated patterns (demand and DG) of each feeder cluster. The daily aggregated pattern demand $p_{d,r}(t)$ is the aggregation of the active power demand of all the customers connected to the reference feeder. Meanwhile, the daily aggregated pattern generation $p_{g,r}(t)$ is the aggregation of the active power generation of all the DG units connected to the reference feeder. Therefore, the model input vector $\mathbb{X} \in \mathbb{R}^{2\hat{k} \times 1}$ represents the demand and generation conditions, at the instant of time t , of the whole LV distribution area organised in $2\hat{k}$ RPs.

3.4. DNN output

The DNN model output $Y \in \mathbb{R}$ (which constitutes the output layer), corresponds to the total technical power loss $p_{loss,T}$ of the whole large-scale LV smart grids as indicated in (30) and it is obtained as the result

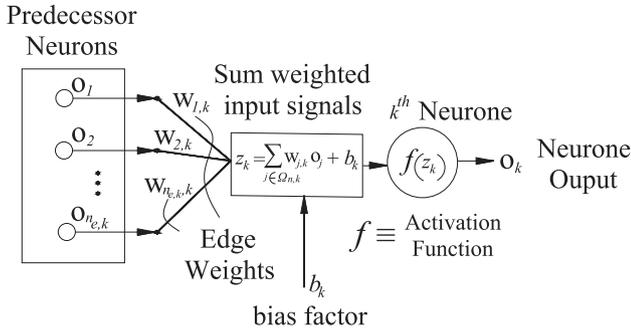


Fig. 6. Data communication between sequential layers.

of a mapping function \mathcal{F}_N of the model configuration \mathbb{W} (edge weights matrix) with the model input vector \mathbb{X} .

$$\mathbb{Y} = \mathcal{F}_N(\mathbb{W}, \mathbb{X}) = p_{loss,T} \quad (30)$$

\mathbb{W} represents the edge weighting matrix in such a way that the matrix element (j, k) , denoted by $w_{j,k}$ represents the weight associated with the edge that connects the j^{th} neurone with the k^{th} neurone. The predecessor neurones of k^{th} neurone are defined as those that are connected to it but that are placed in the previous layer. The set of predecessor neurones of the k^{th} neurone is denoted as $\Omega_{n,k}$, where $n_{e,k}$ is the number of predecessor neurones of the k^{th} neurone.

Each neurone of the model receives real-valued signals from predecessor neurones and produces a new real-value activation signal which is re-sent through the output edges of each neurone, as illustrated in Fig. 6.

The k^{th} neurone (\mathcal{G}_k) processes the signal (z_k) defined in (31), which is the sum of the outputs of the $n_{e,k}$ predecessor neurones (o_j) multiplied by the corresponding edge weights ($w_{j,k}$) plus a bias factor (b_k), which controls the activation process.

$$z_k = \sum_{j \in \Omega_{n,k}} w_{j,k} o_j + b_k \quad (31)$$

Subsequently, the output of the k^{th} neurone, defined as (o_k), is calculated using the activation function $f(z_k) \in (0, 1)$ as indicated in (32). Although a wide collection of activation functions are available in the machine learning literature (e.g., linear, piecewise-linear, threshold, hyperbolic tangent or sigmoid [42]), the sigmoid function has been selected in this paper because it is a smooth s-shape and bounded function.

$$o_k = f(z_k) = \frac{1}{1 + e^{-z_k}} \in (0, 1) \quad (32)$$

3.5. Model training

Model training consists of adjusting the edge weights of matrix \mathbb{W} by using a training dataset in such a way that the trained model performs satisfactorily; when provided with a batch π of input data (demand and generation), the model provides a power loss output $\hat{p}_{loss,T}^{(\pi)}$ that is close enough to the target power loss output $p_{loss,T}^{(\pi)}$ of the large area.

This paradigm used to train the model is called supervised learning [43]. With this method, the active power loss output obtained from the input demand and generation conditions will implicitly contain information about the problem domain, which in this case, is the power flow relations [44].

The most widely used technique to update the edge weights of the model is Back-Propagation (BP) [38], which is a gradient descent-based algorithm that enables the model to be trained to perform a specific task through the calculation of the edge weights values. BP can minimise the overshooting as the gradient get shallower. In this case, BP minimises a loss function denoted by \mathcal{L} using gradient descent. The selection of the

loss function depends on both the learning paradigm (supervised, unsupervised, or reinforcement) and the selected activation function [42]. A widely-used loss function in supervised learning is the mean-squared error which consists of minimising the average squared error between the output power losses $\hat{p}_{loss,T}^{(\pi)}$ and the target power losses $p_{loss,T}^{(\pi)}$ for the π scenario as indicated in (33).

$$\mathcal{L} = \frac{1}{\nu} \sum_{\pi=1}^{\nu} |\hat{p}_{loss,T}^{(\pi)} - p_{loss,T}^{(\pi)}|^2 = \mathcal{L}(w_{i,j}) \quad (33)$$

where ν is the batch size, i.e., the number of samples that are used until the matrix weights are updated as indicated in (34).

$$\Delta w_{i,j} = w_{i,j} \left(\hat{t} + 1 \right) - w_{i,j}(\hat{t}) = \eta \frac{\partial \mathcal{L}}{\partial w_{i,j}}, \forall w_{i,j} \in \mathbb{W} \quad (34)$$

where η is the learning rate and \hat{t} is a time discrete parameter that indicates the iteration step. The learning rate η hyper-parameter influences the speed and quality of learning. Its adjustment is fundamental, as a high value can boost the learning process, but the risk of obtaining suboptimal solutions increases. However, too-low values can lead to a more accurate result at the expense of longer training operations.

Eq. (34) reveals that the sign of the gradient ($\partial \mathcal{L} / \partial w_{i,j}$) indicates how the error varies. The learning process can be regulated through the value of η , ensuring that the model learns from the data minimizing the influence of outliers and/or noise.

The model training algorithm comprises the following steps:

1. **Weight matrix initialisation:** Initially ($\hat{t} = 0$), all the edge weights $w_{i,j}(\hat{t})$ of the weighting matrix $\mathbb{W}(\hat{t})$ are randomly assigned as indicated in (35).

$$\mathbb{W}(\hat{t}) \sim U(w_{i,j}(0), \overline{w_{i,j}(0)}) \quad (35)$$

where $U(\cdot)$ is the uniform distribution, and $w_{i,j}(0)$ is the lower bound for the initialisation of the weights, and $\overline{w_{i,j}(0)}$ is the upper bound.

2. **Propagation of errors:** The first batch of input samples $\mathbb{X}^{(\pi)}$ (demand and generation), $n_\nu = 1$, is sent to the model and propagates forward through the sequential layers until it reaches the output layer, where the corresponding output power losses $\hat{p}_{loss,T}^{(\pi)}(\hat{t})$ are obtained. This sequence of output power losses is compared with the target power losses $p_{loss,T}^{(\pi)}$ (by power flow solutions) and the mean-squared output error $\epsilon_o^{(\pi,\hat{t})}$ is computed using (36).

$$\epsilon_o^{(\pi,\hat{t})} = \frac{1}{\nu} \sum_{i=1}^{\nu} \left| \hat{p}_{loss,T}^{(i)}(\hat{t}) - p_{loss,T}^{(i)} \right|^2 \quad (36)$$

The output error is backward-propagated from the output layer to the previous layers to obtain the error of the hidden layer L_k , $\forall k \in (1, \dots, h)$, defined as $\epsilon_k^{(\pi,\hat{t})}$, which reflects the contribution of that layer to the output error. This is calculated by the product between the output error and column $W_k(\hat{t}) \subset \mathbb{W}(\hat{t}) = [w_{1,k}(\hat{t}), \dots, w_{n_k,k}(\hat{t})]^T$ (n_k is the number of neurones of layer L_k) of the edge weight matrix $\mathbb{W}(\hat{t})$ corresponding to the hidden layer h as indicated in (37).

$$\epsilon_k^{(\pi,\hat{t})} = W_k(\hat{t}) \epsilon_o^{(\pi,\hat{t})}, \forall k \in (1, \dots, h) \quad (37)$$

The error of the hidden layer, $\epsilon_k^{(\pi,\hat{t})}$, is a column vector ($n_k \times 1$) which contains the error contribution of each neurone of the hidden layer.

3. **Gradient descent search:** The gradient of the loss function (which is the slope of the loss function with respect to the edge weight value) is obtained by applying the chain rule to Eq. (33), resulting in Eq. (38) [45].

$$\frac{\partial \mathcal{L}}{\partial W_k(\hat{t})} = -\epsilon_k^{(\pi, \hat{t})} \cdot O_k (I_{n_k} - O_k) \cdot O_{k-1}^T, \quad \forall k \in (1, \dots, h) \quad (38)$$

where

- $O_k = [o_1, \dots, o_{n_k}]$ is the vector of a neurone's outputs of the layer L_k ,
- I_{n_k} is a unit vector of size $(1 \times n_k)$.

4. **Weights matrix update:** The weights are updated using gradient descent with consideration to a learning rate η . The updated edge weights column $W_k(\hat{t} + 1)$ of layer L_k is calculated as (39).

$$W_k(\hat{t} + 1) = W_k(\hat{t}) - \eta \frac{\partial \mathcal{L}}{\partial W_k(\hat{t})}, \quad \forall k \in (1, \dots, h) \quad (39)$$

5. **Output recalculation:** When the entire weighting matrix $\mathbb{W}(\hat{t} + 1)$ has been updated for the first batch, the next batch of samples ($N_v = 2$) is sent to the model and is propagated through it, obtaining a new sequence of output power loss values. The output ends when the difference between the actual error and the previous error is below a threshold error.

$$\begin{aligned} \epsilon_0^{(\pi, \hat{t}+1)} - \epsilon_0^{(\pi, \hat{t})} < \epsilon^{max} &\Rightarrow \text{Ok} \\ \epsilon_0^{(\pi, \hat{t}+1)} - \epsilon_0^{(\pi, \hat{t})} > \epsilon^{max} &\Rightarrow \text{Goto3} \end{aligned} \quad (40)$$

3.6. Training dataset

The training dataset is composed of a sequence of known inputs (i.e., known demand and generation conditions for each representative feeder) and a sequence of known outputs (i.e., the corresponding power losses associated with those demand and generation conditions) which are calculated through power flows [46].

Both input and output training sets constitute the training dataset Ω_T as indicated in (41).

$$\Omega_T := \Omega_{T,x}^{(1)} \cup \dots \cup \Omega_{T,x}^{(K)} \cup \Omega_{T,p}^{(1)} \cup \dots \cup \Omega_{T,p}^{(K)} \quad (41)$$

A cross-validation (CV) procedure is applied to asses the performance of the trained model [47]. In the machine learning field, CV is often commonly performed at the same time in data pipelines as hyper-parameter tuning, which is the process of selecting the values for a model's hyper-parameters that maximise the accuracy, Λ , of the model defined through the absolute percentage error (APE) index [48], shown in Eq. (42).

$$\Lambda = 1 - APE = 1 - \frac{|p_{loss,T} - \hat{p}_{loss,T}|}{p_{loss,T}} \quad (42)$$

where $\hat{p}_{Loss,T}$ is the value of active power loss estimated by the DNN model and $p_{loss,T}$ is the known active power loss value obtained by power flow solutions (balanced/unbalanced).

The tuning process of the model hyper-parameters involves determining the combination of model hyper-parameter values that produce the most accurate model (maximizing Λ). To this end, a collection of n_{hp} different candidate values is defined for each hyper-parameter (Table 3). For instance, for the particular case of the hyper-parameter "number of hidden layers h ", its candidate values are uniformly distributed between a minimum of two hidden layers and a maximum of twenty hidden layers.

The optimal number of epochs, ϑ , is selected by applying the "early stopping" technique which consists of increasing the number of iterations for training the model as long as the accuracy (Λ) continuously increases.

Validation is well known to be often performed by dividing the data into training, test, and validation sets [49]. However, in some cases, this strategy leads to suboptimal hyper-parameters [50]. In this paper, the

Table 3

Candidate values of the hyper-parameters' model for \mathcal{K} -fold process.

Hyper-parameter	Candidate values				
No. Hidden Layers (h)	2	3	5	10	20
No. Hidden neurones (n_h)	4	6	8	12	16
Dropout (ζ)	0.05	0.10	0.20	0.25	0.5
Learning rate (η)	1e-6	1e-4	1e-3	1e-2	1e-1
Batch size (ν)	0.1 ζ	0.25 ζ	0.5 ζ	1.0 ζ	2 ζ
Upper edge-weight $\overline{w_{i,j}(0)}$	0.6	0.7	0.8	0.9	1.0
Lower edge-weight $\underline{w_{i,j}(0)}$	0.1	0.2	0.3	0.4	0.5

\mathcal{K} -Fold technique is used, which has been observed to produce a more accurate model because the variability of the dataset is completely considered. This technique consists of splitting the training dataset into \mathcal{K} subsets as follows:

- Initially, `split 1` is formed by one test fold (Fold 1), subsets ($K = 1$), and the remaining subsets are used as training dataset $K = (2, \dots, \mathcal{K})$. The $n_{hp}^{n_{hp}}$ hyper-parameter combinations are explored for the training dataset to determine the most accurate model. This is performed by building a model with each hyper-parameter combinations, and then training the model with the training dataset of `split 1`. Each model is tested with the test dataset (Fold1), obtaining the power loss accuracy given by (42). From all the hyper-parameter combinations, the one that is selected produces the highest accuracy. The optimal hyper-parameter combination values, denoted by $\gamma^{(1)}$, are retained and the process continues with the next split, `split 2`.
- In `split 2`, the next fold is taken as test dataset $K = 2$ and the first fold and the remainder are included in the training dataset $K = (1, 3, \dots, \mathcal{K})$. For this split, the grid search operation is repeated to obtain the second hyper-parameter combination values $\gamma^{(2)}$ that produce the most accurate model.
- The process is repeated with the upcoming splits until `split \mathcal{K}` , obtaining \mathcal{K} hyper-parameter combination values associated with \mathcal{K} accuracy values.
- From those accuracy scores, the optimal hyper-parameter combination is selected as the combination that provides the highest power loss accuracy Λ . The process is illustrated in Fig. 7.

Note that the selection of the number of \mathcal{K} -folds is not apparent. The aim is that the number of \mathcal{K} -folds must be selected in such an order as to be statistically representative of the problem to solve. A value of $\mathcal{K} = 4$ is very typical in the field of applied machine learning [51].

4. Case study

In this section, the proposed methodology for power loss estimation has been applied to an existing large-scale LV smart grid. The LV distribution area under investigation is located in Madrid (Spain) and the principal characteristics are indicated in Table 4. The distribution area is composed of 147 SS with 1,256 feeders (so $n = 1265$). There are 30,429 residential and commercial customers with a total contractual power of 546 MW. In this area, the contractual DG penetration level is 55% and they are based primarily on rooftop PV panels, which are the typical DG source in the Madrid region. The average smart meter deployment is 88%, which provides hourly measurements from tele-metered customers. The hourly power demand is modelled as a stochastic Markov process, as detailed in [27], to consider the connection on non-telemetered customers. From the utility company, the average power phase unbalance due to the unevenly distributed single-phase customers has been observed to be 13.8%. The geographical area corresponds to a whole metropolitan ZIP code which covers 605 ha.

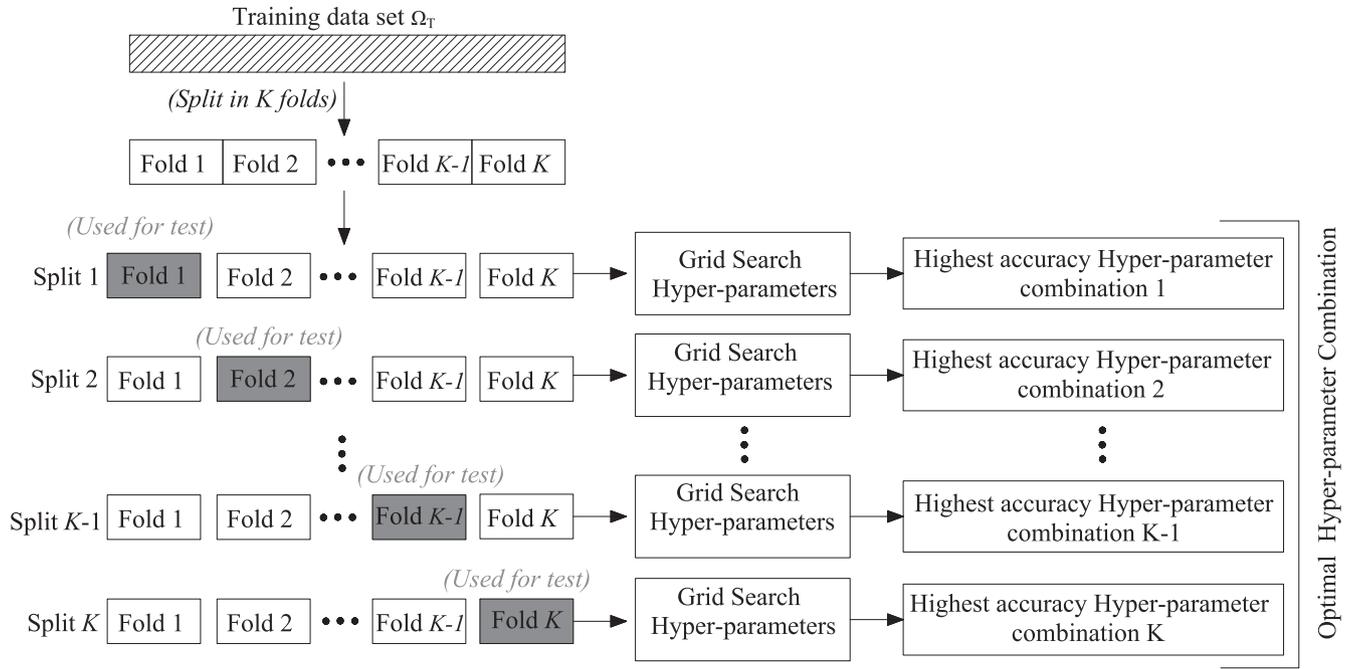


Fig. 7. K -fold cross-validation and hyper-parameter tuning procedure.

Table 4

Case study data.

Case study property	Value
Geographical Area Covered (ha.)	605
No. Secondary Substations (No. SS)	147
No. Customers (three phase, single-phase)*	30429
No. Feeders (n)	1256
Power Contracted (MW)	546
Accumulated feeder Length (km)	273
Max. (PV-based) DG-presence level (%)	55
Aver. Smart Meter penetration level (%)	88
Aver. Power Phase Unbalance (%)	13.8
Aver. Ratio Customers/SS	207
Aver. Ratio Customers/Feeder	24
Network Type	100% Urban
Cables Material	100%Aluminium
Cables impedance	(0.1, 3) Ω /km
Network Configuration	80%Underground 20%Overhead

*residential and commercial.

4.1. Selection of the dataset

Note that the whole area offers wide variability in many parameters, such as feeder properties, customer and DG variability, smart meter deployment, and phase unbalanced.

For that reason, the dataset must be selected to represent a high network diversity but focus on the more relevant information from the point of view of network power losses. Fig. 8 shows the histograms of the \bar{p} feeders characteristics (according to Table 1), and the PDF is indicated with a red line. The feeder characteristics can be observed to follow three different PDFs:

- The characteristics related to power consumption ($X_{1,a}, X_{1,b}, X_{1,c}, X_{9,a}, X_{9,b}, X_{9,c}, X_{10,a}, X_{10,b}, X_{10,c}, X_{11,a}, X_{11,b}, X_{11,c}, X_{12,a}, X_{12,b}$ and $X_{12,c}$) present a Gaussian distribution.
- Meanwhile, the characteristics related to feeder properties (X_2, X_4, X_7, X_8 and X_{13}) are left-skewed.
- The other characteristics such as smart meter penetration (X_3), DG

spatial distribution (X_5), and customers per phase ($X_{14,a}, X_{14,b}, X_{14,c}$) are right-skewed.

From Fig. 8, the following conclusions can be deduced:

- Phase A is observed to be more loaded than the other phases. Consequently, the large LV distribution area is characterised by an unbalanced operation ($X_{1,a}, X_{1,b}, X_{1,c}$).
- The feeder loading level characteristic X_2 varies from 5 to 75%, which means that some feeders are heavily loaded (75% of the power rating of the transformer), and others have significantly reduced loading (5% of the power rating of the SS transformer). The histogram reveals that the PDF is tilted to the left side. Thus, the power contracted by the customers is located primarily between 10 and 50% of the power transformer rating (because the feeder load level is defined as the ratio between power contracted and the power rating of the SS transformer).
- The smart meter deployment (characteristic X_3) varies from a minimum of 55% to the maximum, which is 100%.
- The DG penetration level, which corresponds to the characteristic X_4 , is tilted to the left in such a way that the majority of the feeder has a DG penetration of 50% (ratio between power peak installed and power contracted). Related to this, the self-consumption ratio (X_7) indicates that over half the feeders have a self-consumption ratio less than or equal to 50%.
- The spatial distribution of DG units, X_5 units, and customer's X_6 are very close because the DG facilities are often allocated at the same customer's connection point (PV rooftop facilities).

Fig. 9 shows the correlation matrix of the feeder characteristics. The feeder characteristics related to load unbalance can be observed to be correlated positively ($X_{1,a} - X_{1,b}$) and negatively ($X_{1,b} - X_{1,c}$), which means that when more loading is added to one phase, it is reduced the loading at other phases. Otherwise, the phases will be balanced.

The feeder characteristics ($X_2 - X_3$) and ($X_7 - X_{13}$) are also positively correlated. Because smart meter installation are mandatory in Spain smart for customers with contracted power equal to or less than 15 kW, highly loaded feeders are expected to be prone to have customers with higher contracted power. In addition, note that highly loaded

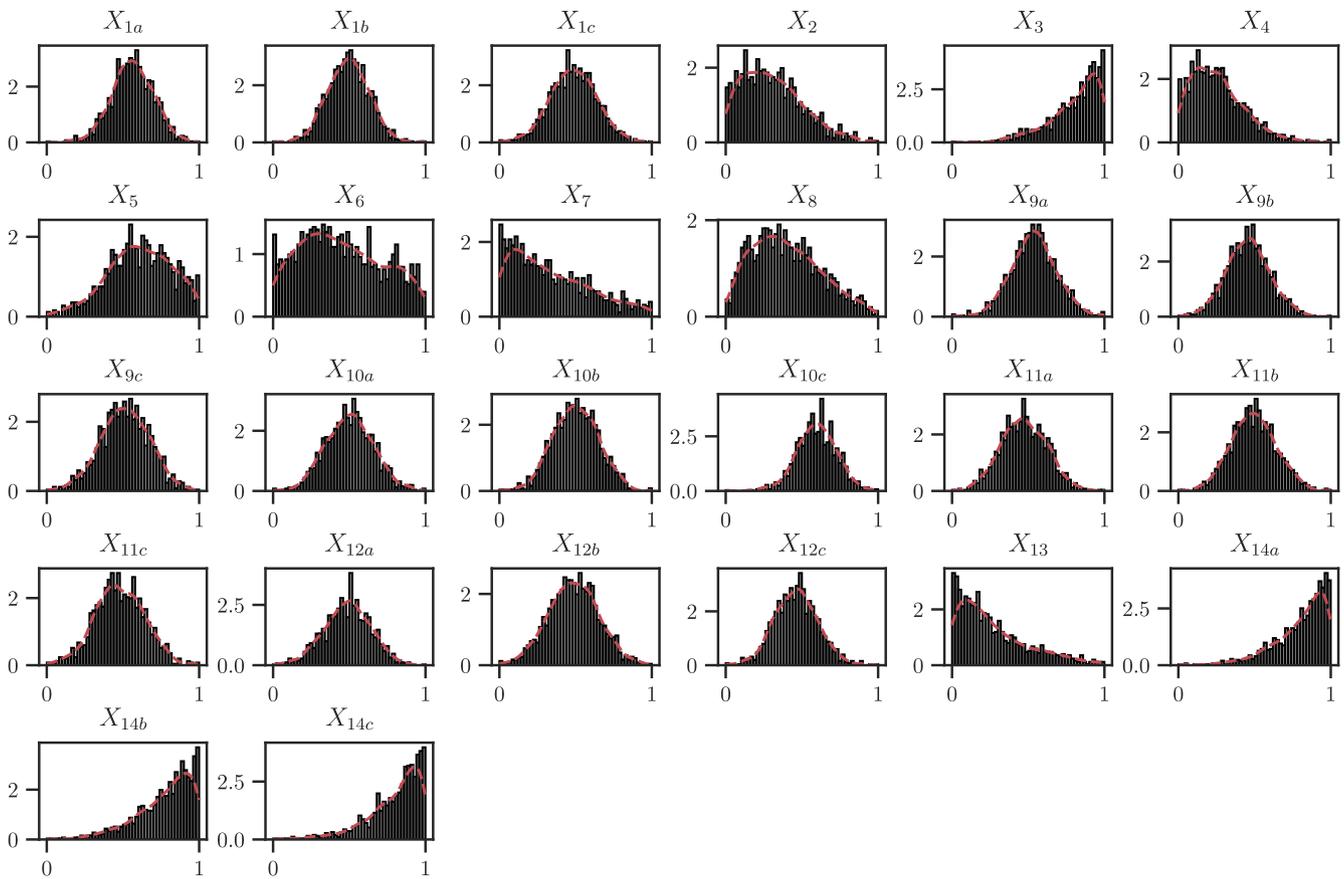


Fig. 8. PDFs and histograms of the feeder characteristics.

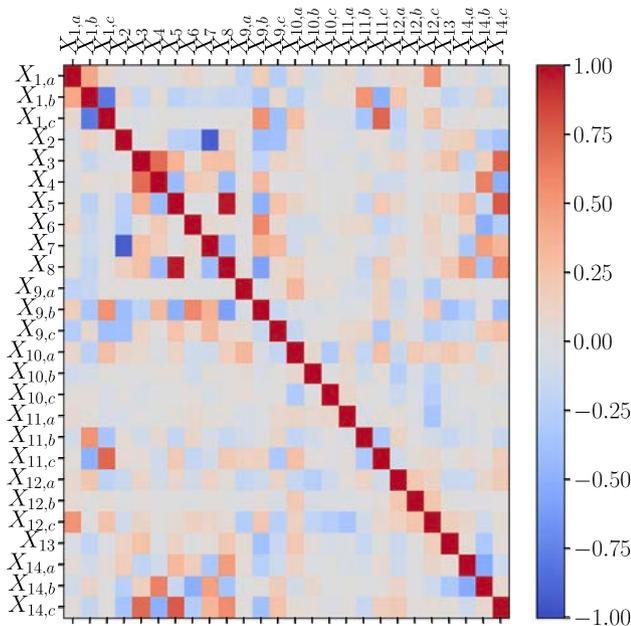


Fig. 9. Correlation matrix of the feeder's characteristics.

feeders presented high correlation self-consumption ratios.

4.2. Data normalisation

Fig. 8 indicates that some feeder characteristics provide very high values (X_{12}) and others present small (X_6). A normalisation process is performed to give the same weight to all feeder characteristics. In this

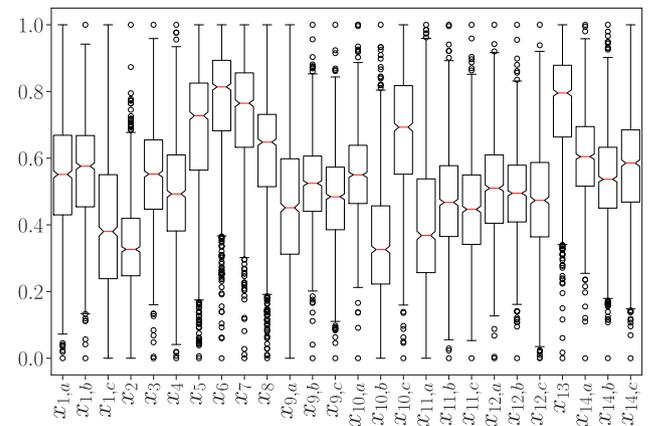


Fig. 10. Boxplots of the normalised feeder characteristics.

case, the feeder characteristics are normalised to vary in the range from 0 to 1. The box plot of each normalised feeder characteristic is obtained and is shown in Fig. 10. Each box plot indicates the median value (red medium vertical line in the box) and the inter-quartile range defined by the upper limit, indicating the 75% percentile (Q_3) and the lower limit, indicating the 25% percentile (Q_1). Moreover, extremes values are shown and are considered outliers. Note that the normalisation process maintains the statistical behaviour of the feeder characteristics, enabling the comparison between them. The feeder properties that exhibit a clear normal distribution (i.e., $X_{1,a}$, $X_{1,b}$, and $X_{1,c}$) have the inter-quartile range around the median zero. The feeder characteristics that have skewed statistical behaviour maintain the majority of the feeder samples in the extremes (for instance, DG penetration level X_4).

Because some of the feeder characteristics are correlated (Fig. 9), a

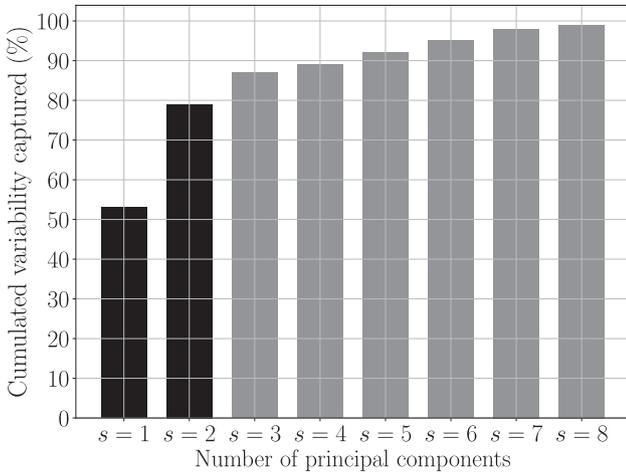


Fig. 11. Cumulated variability captured by the principal component of the PCA analysis.

feature extraction process is performed to obtain a reduced set of uncorrelated feeder features. This is achieved through PCA analysis, as was explained in Section 3.3. The objective of the PCA is to capture the maximum data variability using a reduced number of uncorrelated variables. The PC projections are selected using (9). Fig. 11 shows the cumulated variability captured by each principal component. The first PC ($s = 1$) is observed as capable of capturing 53% of the data variability in the original feeder characteristics. However, this percentage increases if the second projection PC ($s = 2$) is added. The first and second PCs together capture up to 79% of the data variability of the original feeder characteristics. Increasing the number of PC projections can be observed to not notably increase the capture of data variability ($\approx 10\%$ each additional PC projection added).

Consequently, for this case, the PCA analysis enables the reduction of the normalised feeder data matrix to the two-dimensional feeder data matrix, considerably reducing the computational burden but maintaining up to 79% of the original data variability.

4.3. Clustering and representative feeder selection

The $n = 1256$ feeders of the two-dimensional feeder data matrix $\tilde{M}^{n \times s}$ are classified into feeder clusters through the K-means++ procedure explained in Section 2.4. The selection of the optimal number of feeder clusters \hat{k} is not always straightforward but emanates from a compromise between a reduced number of clusters and a large value of the GS. The K-means++ algorithm has been executed 1,000 times using different initial centroid seeds.

Fig. 12 shows the evolution of the GS index as the number of clusters increases (red line), and the percentage of “variance explained” (blue line), which is calculated using the F-test [52]. Note that for $\hat{k} = 2$ clusters, a maximum value of $GS_{\hat{k}=2} = 0.48$ is achieved. However, the percentage of “variance explained” by two clusters is very low (40%). Consequently, a compromise value of $\hat{k} = 4$ feeder clusters provides a good percentage of “variance explained” [52] and a good GS value [12,13].

Fig. 13.a) shows the scatter plot of the feeder data for each of the $\hat{k} = 4$ feeder clusters. Within each feeder cluster, a representative feeder \tilde{c}_r is selected as the feeder closest to the centroid’s cluster. Finally, Fig. 13.b) shows the silhouette index value (23) for the selected clustering configuration (i.e., $\hat{k} = 4$ feeder clusters). The vertical red line indicates the silhouette index value for the selected clustering configuration. The figure indicates that only a few feeder samples are below the zero limit.

Table 5 shows the Euclidean distance factor α_r (used to each representative feeder input in the model) and the number of feeder

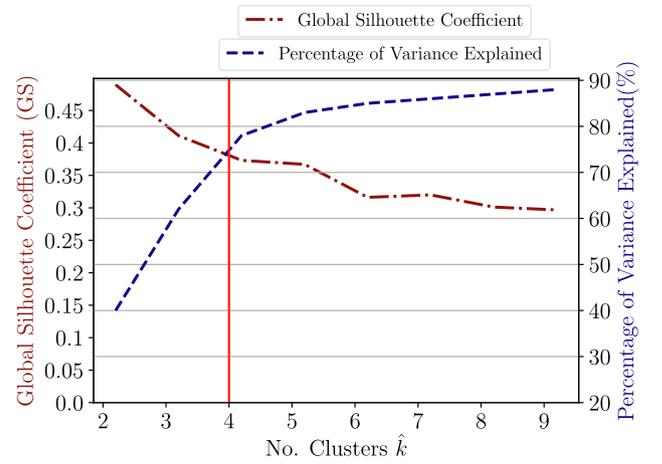


Fig. 12. Selection of the optimal number of clusters.

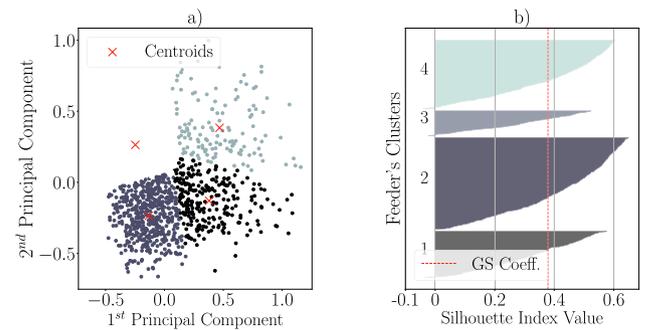


Fig. 13. Feature extraction and clustering results: (a) Scatter plot of the feeder clusters obtained through first and second PCs, (b) Silhouette index value plot for $\hat{k} = 4$ feeder clusters.

Table 5

Euclidean distance factor per feeder cluster and the number of feeders per cluster.

Representative Feeder \tilde{c}_r	Euclidean distance factor α_r	No. feeders per cluster n_r
$r = 1$	1.43	314
$r = 2$	1.64	287
$r = 3$	1.75	345
$r = 4$	1.82	310

clusters assigned to each cluster.

4.4. Model training

The deep learning loss model presented in this paper has been trained and validated using the Python programming language and the scikit-learn library [53]. The training dataset Ω_T has been synthetically produced using the smart meter data of the Spanish large-scale LV distribution area gathered over the period from 2014–2018 with a time resolution of 10 min. Moreover, the generation data from each of the DG units for the same period is also available.

For each reference feeder, a collection of APs for the demand and generation has been synthetically created based on yearly smart meter measurement data. These consist of estimating the PDFs of both the demand and distributed generation using the Kernel Density Estimation method for each representative feeder. The PDFs obtained exhibit a shape similar to the Gaussian distribution but with considerable asymmetry, in the form of positive skew (tilted left). From the PDFs obtained for each representative feeder, the first and second statistical moments (expected value and the variance) are sampled in a Monte

Table 6

Synthetic demand and generation APs formulas. $\omega \in (1, \dots, \kappa)$ is the index for demand and generation scenarios (which is a daily AP) and κ is the total number of daily APs of each type synthetically generated.

Demand APs		
Winter	WD	$D_1: P_{d,r}^{wi+wd(\omega)}(t) = \mu_{P_{d,r}^{wi+wd}(t)} \pm \sigma_{P_{d,r}^{wi+wd}(t)}$
	NWD	$D_2: P_{d,r}^{wi+nwd(\omega)}(t) = \mu_{P_{d,r}^{wi+nwd}(t)} \pm \sigma_{P_{d,r}^{wi+nwd}(t)}$
Summer	WD	$D_3: P_{d,r}^{su+wd(\omega)}(t) = \mu_{P_{d,r}^{su+wd}(t)} \pm \sigma_{P_{d,r}^{su+wd}(t)}$
	NWD	$D_4: P_{d,r}^{su+nwd(\omega)}(t) = \mu_{P_{d,r}^{su+nwd}(t)} \pm \sigma_{P_{d,r}^{su+nwd}(t)}$
Generation APs		
Winter		$G_1: P_{g,r}^{wi(\omega)}(t) = \mu_{P_{g,r}^{wi}(t)} \pm \sigma_{P_{g,r}^{wi}(t)}$
Summer		$G_2: P_{g,r}^{su(\omega)}(t) = \mu_{P_{g,r}^{su}(t)} \pm \sigma_{P_{g,r}^{su}(t)}$

Carlo process. The standard deviation is used instead of the variance (obtained as the square root of the variance) for dimensional analysis. Subsequently, with every expected value and standard deviation samples, an aggregated pattern is formed for each representative feeder. The expected value represents the demand-generation trend value of the aggregated pattern ($\mu_{(\cdot)}$), while the standard deviation ($\sigma_{(\cdot)}$) represents the uncertainty component level. A large number of synthetic APs $\kappa > 1000$ are produced using this process. Consequently, these demand and generation APs represent the aggregation of all the customers connected to the reference feeder (in the case of demand APs), and the aggregation of all the DG units connected to the reference feeder (in the case of generation APs).

Moreover, the variability of demand and generation throughout the year (seasons, working days, holidays, etc.) are considered by discriminating the synthetically produced APs for the winter and summer seasons and for working-days (WD) and non-working days (NWD) (holidays and weekends), as indicated in Table (6).

Each scenario has been simulated in every representative feeder, and the corresponding active power losses have been obtained by solving a power flow problem (balanced/unbalanced) (Fig. 14) [46].

Fig. 15 shows a daily sample of the training data produced.

4.5. Model Validation

For the \mathcal{K} -fold cross-validation procedure, a number of $\mathcal{K} = 4$ folds have been selected to split the training dataset [51]. Table 7 shows the hyper-parameter values associated with the highest accuracy score trajectories obtained. The model with the higher accuracy corresponds to \mathcal{K} -Fold 1, where the architecture of the model consists of four layers: the input layer has eight input signals (aggregated demand and

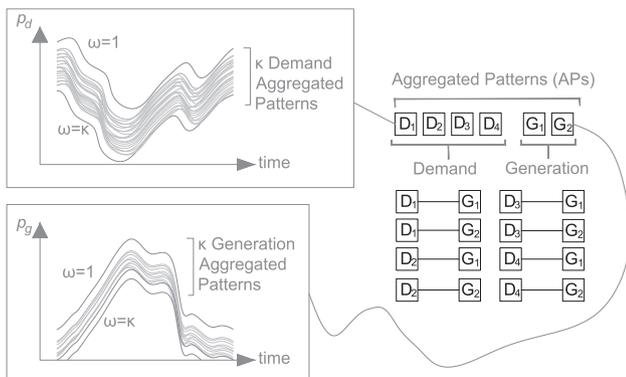


Fig. 14. Combination of demand APs and generation APs for one representative feeder.

aggregated generation for each cluster); the first hidden layer has six neurones; the second hidden layer has four neurones; and, finally, the output layer generates the total power losses of the large-scale area. A predominant batch size of 240 can be observed to provide the model with the highest accuracy. The final architecture is shown in Fig. 16.

4.6. Model results

When the deep learning loss model has been trained and validated, it can be applied for the estimation of the technical losses in the large-scale LV smart grid for a 24-h period with a resolution of 10 min. Fig. 17 shows the loss estimation on a daily basis for the whole large area. Fig. 17a) shows the aggregated power demand, Fig. 17b) shows the aggregated power generation from the PV-based units of each representative feeder. Fig. 17c) shows the loss estimation using the deep learning loss model for the large-scale LV distribution area (solid line); the dashed line shows the losses obtained by an unbalanced power flow. Finally, Fig. 17d) shows the accuracy of the model over the day, which results in an average value of 88%, outperforming the results obtained in [23,21].

Fig. 18 shows the power loss estimation for the unbalanced large area by applying the deep learning model proposed in this paper for a period of four weeks (solid line), and the exact power losses obtained with the unbalanced power flow solution (dashed line); note that the power flow requires the exact network topology and customer/generation data. Furthermore, note that the overall accuracy of the deep learning model over a four-week period is 87% compared with the exact power flow solution. Moreover, the computational speed of the deep learning model is approximately 10 times lower than that of the traditional approach (power flow); for this case study, the power flow requires 1 h and the DNN 7 min.

4.7. Comparative results

As mentioned in the introduction to this paper, few methodologies have been proposed in literature to evaluate the power losses in LV large-scale smart grids, considering both unbalanced operation and DG. In this section, two different network situations are considered:

(a) Balanced network

To provide a comparative result, the proposed deep learning loss model using out-of-the-box demand and generation data is compared with the loss estimation method proposed by [54]. Note that the method in [54] cannot be applied to unbalanced networks; hence the LV distribution area under study (Table 4) has been modified to represent a fully balanced three-phase network where single-phase customers are uniformly distributed among three different phases, which means zero unbalance.

Fig. 18 shows the comparison of loss estimation for large LV distribution areas (balanced network situation) using the proposed deep learning loss model (solid line), the real losses using deterministic balanced power flow (dark dashed line), and the loss estimation method proposed by [54] (red line). The upper plot indicates that the deep learning method outperforms the results of [54], which, due to the lack of an accurate consideration of DG generation, provides negative values of power losses at certain times of the week. The lower plot of the same figure shows the APE results of both methods compared with the exact power flow results. The deep learning method can be observed to provide minimal error values, demonstrating its superior accuracy.

(b) Unbalanced network

In this second situation, single-phase customers are unevenly distributed among the different phases producing an average power unbalance of 13.8% among the different phases. The proposed deep learning loss model, considering phase unbalance, is compared with the unbalanced power flow to quantify power losses in the large scale LV unbalanced area. Fig. 19 shows the real network losses provided by the unbalanced power flow [46] in dashed lines and the aggregated

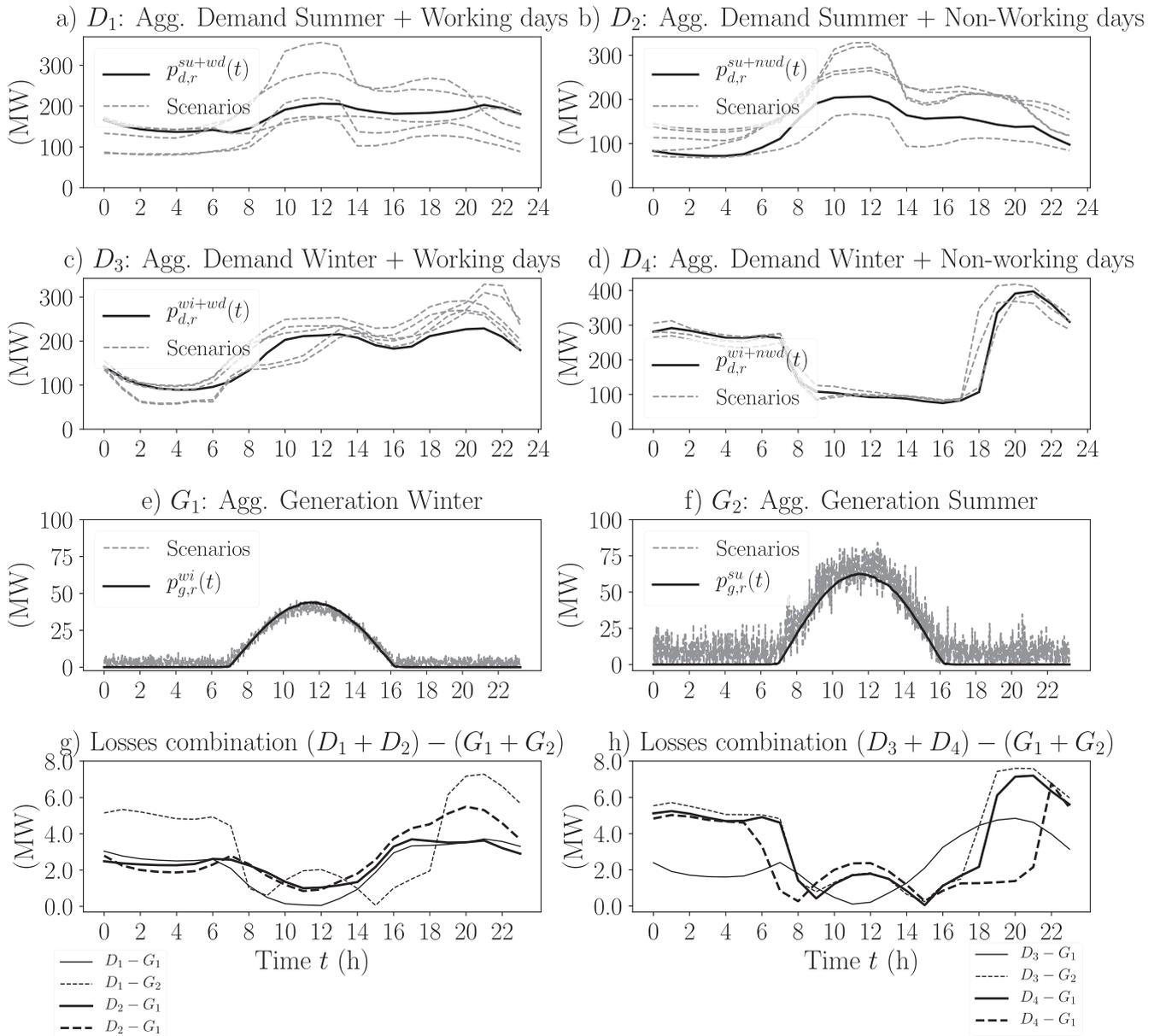


Fig. 15. Training data for the deep learning losses model.

network losses provided by the proposed DNN loss model in a solid dark line. Fig. 19.b indicates that the absolute percentage error is below 0.1% for the 99% of the period analysed.

5. Conclusion

This article presents a generic deep learning loss model to estimate the technical losses in large-scale LV smart grid with DG penetration and unbalanced operation. The proposed loss model is based on a Deep Neural Network that has been specifically formulated as a power loss

estimator for large-scale LV distribution areas considering the LV distribution network’s variability, such as: load unbalance, power variability from distributed generation and uncertainty in smart meters measurements. Due to the great diversity in network characteristics, customer demand, and DG generation, some previous stages have been considered. First, to train the DNN loss model, a set of demand-generation scenarios has been synthetically created for the representative feeders, and their corresponding power losses have been calculated using unbalanced power flow. The model has been trained using the \mathcal{K} -Fold procedure, simultaneously obtaining the optimal combination

Table 7
Hyper-parameter combinations with the highest accuracy for each split.

Accuracy Λ	\mathcal{K} Fold	Architecture ($n_1: \dots: n_n$)	Dropout ζ	Learning rate η	Weight initialisation ($w_{i,j}^{min}, w_{i,j}^{max}$)	Batch size ν
0.9365	1	(8:6:4:1)	0.24	1e-2	(0.06,0.75)	240
0.9271	2	(8:6:4:1)	0.32	1e-3	(0.27,0.52)	240
0.9205	3	(8:4:8:1)	0.41	1e-2	(0.35,0.60)	240
0.9200	4	(8:12:12:1)	0.38	1e-4	(0.02,0.72)	48

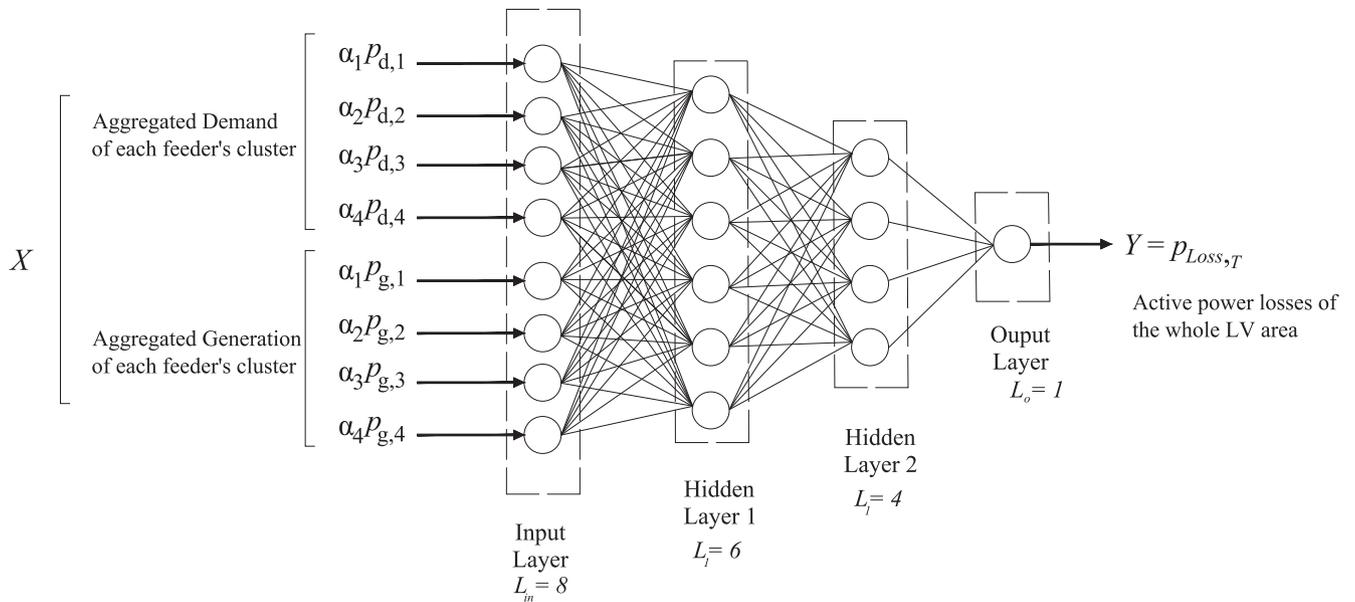


Fig. 16. Final architecture of the DNN model.

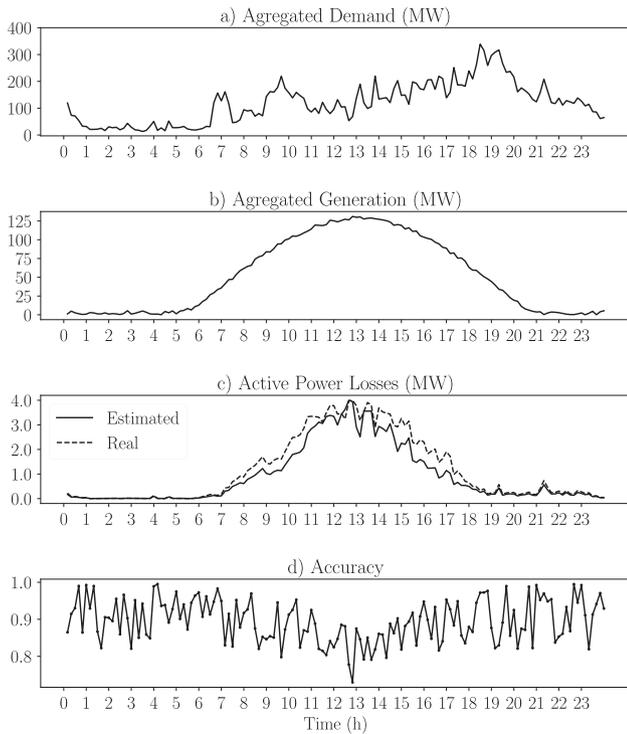


Fig. 17. Active power loss estimation results on one daily basis.

of the model's hyper-parameters. The trained and validated model is subsequently fed with the demand and generation data from the metering infrastructures of the representative feeders, scaled up to the number of feeders belonging to the same feeder cluster. The deep learning loss model has been implemented in an existing LV large-scale distribution area, demonstrating a good performance as a power loss estimator compared with the traditional approach based on deterministic power flow solutions. The obtained results indicate that applying the proposed deep-learning loss model to large-scale LV distribution areas enables DSOs to efficiently quantify technical losses in large areas for management of network losses and to identify network areas where the reduction of power losses must be performed. Furthermore, it can be used as a planning tool to minimise losses in large-scale LV

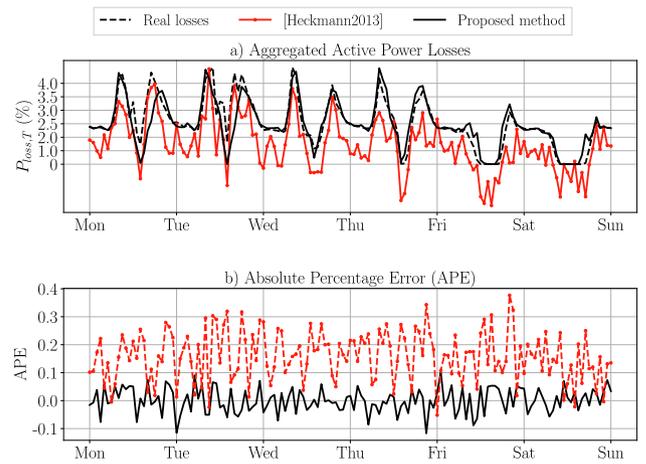


Fig. 18. Comparison of active power loss between the proposed method and the method proposed in [54] under balanced situations (7-13 October, 2019).

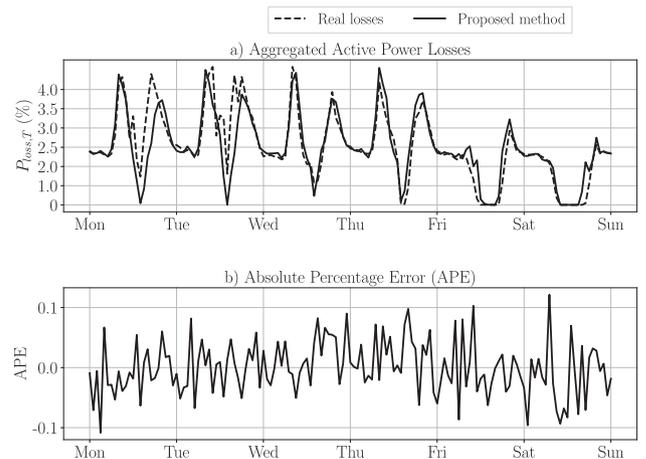


Fig. 19. Comparison of active power losses between the proposed method and the unbalanced power flow (7-13 October, 2019).

distribution areas under uncertain conditions. This paper has demonstrated that the presented deep-learning loss model offers remarkable robustness under uncertainty of input data, making having detailed information about the network topology or accurate load and generation power measurements unnecessary. The robustness of the proposed model for loss estimation in large LV distribution areas with DG penetration and unbalanced operation enables this method to be applied in real-time applications.

Declaration of Competing Interest

None.

Acknowledgement

This work has been partly funded by the Spanish Ministry of Economy and Competitiveness through the National Program for Research Aimed at the Challenges of Society under the project OSIRIS (RTC-2014-1556-3).

Appendix A. Supplementary material

Supplementary data associated with this article can be found, in the online version, at <https://doi.org/10.1016/j.ijepes.2020.106054>.

References

- Messinis GM, Hatzigiorgiou ND. Review of non-technical loss detection methods. *Electric Power Syst Res* 2018;158:250–66.
- Graditi G, Di Silvestre ML, Gallea R, Riva Sanseverino E. Heuristic-based shiftable loads optimal management in smart micro-grids. *IEEE Trans Industr Inf* 2015;11(1):271–80. <https://doi.org/10.1109/TII.2014.2331000>.
- Di Somma M, Graditi G, Siano P. Optimal bidding strategy for a der aggregator in the day-ahead market in the presence of demand flexibility. *IEEE Trans Industr Electron* 2019;66(2):1509–19. <https://doi.org/10.1109/TIE.2018.2829677>.
- Fu X, Chen H, Cai R, Xuan P. Improved lsf method for loss estimation and its application in dg allocation. *IET Gener Transm Distrib* 2016;10(10):2512–9.
- Ibrahim KA, Au MT, Gan CK, Tang HJ. System wide mv distribution network technical losses estimation based on reference feeder and energy flow model. *Int J Electr Power Energy Syst* 2017;93:440–50.
- Urquhart AJ, Thomson M. Impacts of demand data time resolution on estimates of distribution system energy losses. *IEEE Trans Power Syst* 2015;30(3):1483–91. <https://doi.org/10.1109/TPWRS.2014.2349157>.
- Poursharif G, Brint A, Black M, Marshall M. Using smart meters to estimate low-voltage losses. *IET Gener, Transmiss Distrib* 2018;12(5):1206–12.
- Gaunt CT, Namanya E, Herman R. Voltage modelling of LV feeders with dispersed generation: Limits of penetration of randomly connected photovoltaic generation. *Electric Power Syst Res* 2017;143:1–6.
- Ma C, Dasenbrock J, Töbermann JC, Braun M. A novel indicator for evaluation of the impact of distributed generations on the energy losses of low voltage distribution grids. *Appl Energy* 2019:674–83.
- Amaris H, Molina YP, Alonso M, Luyo JE. Loss allocation in distribution networks based on aumann-shapley. *IEEE Trans Power Syst* 2018;33(6):6655–66.
- Dortolina CA, Nadira R. The loss that is unknown is no loss at all: a top-down/bottom-up approach for estimating distribution losses. *IEEE Trans Power Syst* 2005;20(2):1119–25.
- Mateo C, Prettico G, Gómez T, Cossent R, Gangale F, Frías P, et al. European representative electricity distribution networks. *Int J Electr Power Energy Syst* 2018;99:273–80.
- Rigoni V, Ochoa LF, Chicco G, Navarro-Espinosa A, Gozel T. Representative residential lv feeders: a case study for the north west of England. *IEEE Trans Power Syst* 2016;31(1):348–60.
- Usman M, Coppo M, Bignucolo F, Turri R. Losses management strategies in active distribution networks: a review. *Electric Power Syst Res* 2018:116–32.
- Bletterie B, Kadam S, Renner H. On the Classification of low voltage feeders for network planning and hosting capacity studies. *Energies*, vol. 11, 651. <https://doi.org/10.3390/en11030651>.
- Rösch T, Treffinger P. Cluster analysis of distribution grids in baden-württemberg. *Energies*, vol. 12, 20. <https://doi.org/10.3390/en12204016>.
- Hong YY, Chao ZT. Development of energy loss formula for distribution systems using fcn algorithm and cluster-wise fuzzy regression. *IEEE Trans Power Deliv* 2002;17(3):794–9.
- Dashtaki AK, Haghifam MR. A new loss estimation method in limited data electric distribution networks. *IEEE Trans Power Deliv* 2013;28(4):2194–200.
- Fang L, Ma K, Li R, Wang Z, Shi H. A statistical approach to estimate imbalance-induced energy losses for data-scarce low voltage networks. *IEEE Trans Power Syst* 2019;34(4):2825–35. <https://doi.org/10.1109/TPWRS.2019.2891963>.
- Monteiro RV, Guimarães GC, Silva FB, da Silva Teixeira RF, Carvalho BC, Finazzi ADP, et al. A medium-term analysis of the reduction in technical losses on distribution systems with variable demand using artificial neural networks. An Electrical Energy Storage approach. *Energy* 2018;164:1216–28. <https://doi.org/10.1016/j.energy.2018.09.021>.
- Wang S, Dong P, Tian Y. A novel method of statistical line loss estimation for distribution feeders based on feeder cluster and modified xgboost. *Energies* 10. <https://doi.org/10.3390/en10122067>.
- Kang M-S, Chen CS, Lin C-H, Huang C-W, Kao M-F. A systematic loss analysis of taipower distribution system. *IEEE Trans Power Syst* 2006;21(3):1062–8.
- Leal AG, Jardini JA, Magrini LC, Ahn SU. Distribution transformer losses evaluation: a new analytical methodology and artificial neural network approach. *IEEE Trans Power Syst* 2009;24(2):705–12.
- Chen CS, Lin CH, Huang MY, Chen HD, Kang MS, Huang CF. Development of distribution feeder loss models by artificial neural networks. *IEEE Power App Syst* 2005;PAS-1: 164–70.
- Hsu CT, Tzeng YM, Chen CS, Cho MY. Distribution feeder loss analysis by using an artificial neural network. *Electric Power Syst Res* 1995;34:85–90.
- López G, Moreno JI, Amarís H, Salazar F. Paving the road toward Smart Grids through large-scale advanced metering infrastructures. *Electric Power Syst Res* 2015;120:194–205.
- Velasco J, Amarís H, Alonso M, Miguelez M. Stochastic technical losses analysis of smart grids under uncertain demand. in: 2018 53rd International Universities power engineering conference (UPEC). 2018. p. 1–6.
- de Souto MCP, de Araujo DSA, Costa IG, Soares RGF, Ludermitr TB, Schliep A. Comparative study on normalization procedures for cluster analysis of gene expression datasets. *IEEE International joint conference on neural networks (IEEE World Congress on Computational Intelligence)* 2008;2008:2792–8.
- Andrew Watters P, Boslaugh S. *Statistics in a Nutshell*. O'Reilly Media; 2018.
- Jolliffe IT. *Principal component analysis*. Springer; 2002.
- Lu H, Venetsanopoulos A, Plataniotis KN. *Multilinear subspace learning*. CRC Press; 2013.
- Ke-Lin D, S.M.N.S. *Neural networks and statistical learning*. Springer; 2014.
- Rousseeuw PJ. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J Comput Appl Math* 1987;20:53–65.
- Xiang M, Yu J, Yang Z, Yang Y, Yu H, He H. Probabilistic power flow with topology changes based on deep neural network. *Int J Electr Power Energy Syst*, vol. 117. <https://doi.org/10.1016/j.ijepes.2019.105650>.
- Zang H, Cheng L, Ding T, Cheung KW, Wei Z, Sun G. Day-ahead photovoltaic power forecasting approach based on deep convolutional neural networks and meta learning. *Int J Electr Power Energy Syst*, vol. 118. <https://doi.org/10.1016/j.ijepes.2019.105790>.
- Grus J. *Data science from scratch. First principles with python*. O'Reilly Media; 2015.
- Geoffrey YLYB. *Deep learning*. Nature 2015;521:436–44.
- Schmidhuber J. *Deep Learning in neural networks: an overview*. *Neural Networks* 2015;61:65–117.
- Bergstra J, Bengio Y. Random search for hyper-parameter optimization. *J Mach Learn Res* 2012;13:285–305.
- Srivastava N, Hinton AG, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural network from overfitting. *J Mach Learn* 2014;15:1929–58.
- Ehsan A, Yang Q. State-of-the-art techniques for modelling of uncertainties in active distribution network planning: a review. *Appl Energy* 2019;239:1509–23. <https://doi.org/10.1016/j.apenergy.2019.01.211>.
- Simon S H. *Neural networks: a comprehensive foundation*. Prentice Hall; 1999.
- Bishop CM. *Pattern recognition and machine learning*. Springer; 2006.
- Kumar V, Abraham OA, Snel V. Metaheuristic design of feedforward neural networks: a review of two decades of research. *Eng Appl Artif Intell* 2017;60:97–116.
- Rashid T. *Make your own neural network*. Amazon Media EU 2017.
- Velasco JA, Rigoni V, Soroudi A, Keane A, Amaris H. Optimising load flexibility for the day ahead in distribution networks with photovoltaics. *IEEE Milan PowerTech* 2019;2019:1–6. <https://doi.org/10.1109/PTC.2019.8810963>.
- Du K-L, Swamy MNS. *Neural networks and statistical learning*. Springer; 2013.
- Goodfellow I, Bengio Y, Courville A. *Deep learning*. MIT Press; 2016.
- Kuhn M, Johnson K. *Applied predictive modelling*. Springer; 2013.
- Russel S, Norving P. *Artificial intelligence: a modern approach*. Prentice-Hall; 1994.
- Hastie T, Tibshirani R, Witten D, James G. *Introduction to statistical learning: with applications in R*. Springer; 2017.
- Hayter A. *Probability and statistics for engineers and scientists*, Brooks/Cole; 2012.
- Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, et al. *Scikit-learn: machine learning in Python*. *J Mach Learn Res* 2011;12:2825–30.
- Heckmann W, Barth H, Reimann T, Hamann L, Dasenbrock J, Scheidler A, et al. Detailed analysis of network losses in a million customer distribution grid with high penetration of distributed generation. In: 22nd International conference and exhibition on electricity distribution (CIRED 2013); 2013.