applied sciences

MDPI

Article

SmartFog: Training the Fog for the Energy-Saving Analytics of Smart-Meter Data

Michele Scarpiniti *, Enzo Baccarelli, Alireza Momenzadeh and Aurelio Uncini

Department of Information Engineering, Electronics and Telecommunications (DIET), Sapienza University of Rome, via Eudossiana 18, 00185 Rome, Italy; enzo.baccarelli@uniroma1.it (E.B.); alireza.momenzadeh@uniroma1.it (A.M.); aurelio.uncini@uniroma1.it (A.U.)

* Correspondence: michele.scarpiniti@uniroma1.it; Tel.: +39-06-4458-5869

Received: 12 September 2019; Accepted: 3 October 2019; Published: 8 October 2019

Abstract: In this paper, we characterize the main building blocks and numerically verify the classification accuracy and energy performance of SmartFog, a distributed and virtualized networked Fog technological platform for the support for Stacked Denoising Auto-Encoder (SDAE)-based anomaly detection in data flows generated by Smart-Meters (SMs). In SmartFog, the various layers of an SDAE are pretrained at different Fog nodes, in order to distribute the overall computational efforts and, then, save energy. For this purpose, a new *Adaptive* Elitist Genetic Algorithm (AEGA) is "ad hoc" designed to find the optimized allocation of the SDAE layers to the Fog nodes. Interestingly, the proposed AEGA implements a (novel) mechanism that *adaptively* tunes the exploration and exploitation capabilities of the AEGA, in order to quickly escape the attraction basins of local minima of the underlying energy objective function and, then, speed up the convergence towards global minima. As a matter of fact, the main distinguishing feature of the resulting SmartFog paradigm is that it accomplishes the *joint* integration on a distributed Fog computing platform of the anomaly detection functionality and the minimization of the resulting energy consumption. The reported numerical tests support the effectiveness of the designed technological platform and point out that the attained performance improvements over some state-of-the-art competing solutions are around 5%, 68% and 30% in terms of detection accuracy, execution time and energy consumption, respectively.

Keywords: Fog Computing; Smart-Meter; energy efficiency; anomaly detection; Stacked Denoising Auto-Encoder (SDAE); Adaptive Elitist Genetic Algorithm (AEGA)

1. Introduction

Anomalies in Smart-Meter (SM) data, due to a malfunction of the meter or a theft from users, cause a major problem for electric companies [1]. In this regard, in recent years, many research efforts have been devoted to the development of specific computational algorithms that would be able to quickly identify the anomaly with high accuracy. Among the different approaches presented in the literature, the most interesting ones are those based on computational intelligence, such as Deep Neural Networks (DNNs), clustering [1], and those based on linear programming [2].

Recently, promising techniques are developed by exploiting deep learning approaches [3]. In fact, when correctly trained on the available recorded data, these methods are able to attain high accuracy [4]. However, since a deep architecture consists in the cascade of several computational layers, the resulting layered structure usually makes the training phase difficult, due to vanishing gradient problem among others [4]. To overcome this problem, a layer-wise pretraining solution, followed by a fine-tuning of the resultant whole architecture, was proposed in the literature. This method provides good results in practical applications [5].

In all cases, approaches based on deep learning suffer high computational complexity, since very large architectures with a huge number of parameters have to be trained [5]. A (partial) solution to this problem is to resort to the Cloud Computing (CC) paradigm, that enables the ubiquitous access to large-size pools of virtualized computing resources by establishing (typically) multi-hop wireless communication paths [6]. Although CC offers a huge computational capability, it is characterized by its high energy consumption [7].

The quite recent paradigm of Fog Computing (FC) can help to overcome this problem [7,8]. In fact, by definition, FC enables the pervasive local access to distributed and *virtualized* small-size energy-saving pools of computing resources that can be quickly provisioned, dynamically scaled up/down and released on an on-demand basis. Nearby resource-limited edge devices (e.g., SMs) may access these resources by establishing single-hop WiFi-supported communication links [7]. Hence, in principle, virtualized Fog nodes could be exploited in a distributed way to implement the unsupervised pretraining of the layer of a deep architecture. The (possible) supervised final fine-tuning of the whole SDAE can be performed by a CC node, being this phase of centralized type.

However, a potential problem of multi-tier architectures (composed of devices, Fog nodes and Cloud nodes) is the considerable amount of energy wasted by the networking resources needed for inter-node communication. Also in this case, in literature there exist some solutions being able to optimize the total energy consumption, such as the Genetic Algorithm (GA) [9]. Hence, it is reasonable to expect that the exploitation of the Fog architecture, along with the distributed pretraining procedure, could reduce the energy consumption with respect to solutions based on the use of the Cloud only.

Motivated by these considerations, the contribution of this paper is threefold:

1. we introduce a Stacked Denoising Auto-Encoder (SDAE) architecture for the anomaly detection of SM data, and exploit the FC to implement the distributed pretraining of the SDAE layers;
2. we engineer the overall networked Fog infrastructure, in order to minimize the total communication-plus-computing energy wasted by the proposed anomaly detector; and,
3. we design a new *Adaptive* Elitist GA (AEGA) to optimize the mapping of the SDAE layers to be pretrained to the available Fog nodes, in order to meet the aforementioned energy minimization tasks. The adaptive mechanism implemented by the proposed AEGA makes it robust against trapping phenomena generated by the attraction basins of local minima of the underlying objective function and, then, reduces the resulting convergence time.

The rest of the paper is organized as follows. After a review of the related work in Section 2, in Section 3, we present the overall architecture of the SmartFog platform, while, in Section 4, we point out the formal models adopted for the analytical evaluation of the resulting computing and networking energy. In Section 5, we present the main design principles behind the proposed AEGA and, then, we detail the AEGA pseudo-code. Section 6 is devoted to numerically testing the performance of the resulting optimized SmartFog platform in terms of detection accuracy, execution time and energy consumption. For this purpose, the results of performance comparisons against some competing state-of-the-art benchmark solutions are also presented. Finally, in Section 7, we recap the main contributions of this paper and point out some areas for future research.

2. Related Work

It is expected that the convergence of the DNN and FC paradigms will be fostered by the results stemming from (at least) three on-going research lines, namely [10]: (i) the design of anomaly detection algorithms; (ii) the design and optimization of hierarchically organized multi-tier Fog platforms; and, (iii) the development of "ad hoc" algorithms for the supervised distributed training of DNNs.

Anomaly detection algorithms exploiting DNNs—A very challenging issue when dealing with SM data is represented by the early detection of anomalies. In this regard, the literature offers several contributions, like [11–14].

Specifically, the work in [11] is focused on a Bayesian approach for the segmentation of data sequences and the identification of anomalies based on an empirical estimate of the recurrent behavior

of observed changepoints. Paper [12] provides a comparative analysis of some traditional "shallow" machine learning algorithms (like k-NN, MLP and SVM), which were used to classify measurements as regular or anomaly. In this work, well-known batch and online learning algorithms (both supervised and semisupervised) were employed to cope with the anomaly detection problem. Differently from our paper, works in [11,12] do not exploit deep learning and do not take account of the distribution of the detection algorithm over several Fog nodes.

More recently, another line of research is addressed towards deep learning techniques, e.g., see [13,14] and references therein. Specifically, the work in [13] integrates a DNN with a cyber-physical protocol to identify and mitigate the information corruption in sensor-acquired data. The proposed technique exploits unsupervised training based on Deep Belief Networks (DBN). The authors in [14] propose a real-time detection of anomalies in Smart Grid by using an architecture based on Conditional Deep Belief Networks (CDBNs). This work is also characterized by a model that copes with the limited number of available measurements. Differently from our paper, works in [13,14] are based on DBNs that exhibit a slow convergence behavior and high computational costs, while we focus on the use of SDAE that retains a good compromise between convergence speed and computational cost.

Overall, *unlike* our paper, works in [11–14] and references therein do not take account of the accuracy-vs.-energy performance over a distributed and virtualized networked Fog platform, deeply investigated in this paper.

Multi-tier Fog Computing platforms for the support of DNNs—The general topic of the design of multi-tier FC platforms is receiving large attention, mainly due to the emerging areas of the Mobile Cloud, Pervasive Computing and Edge Computing [15,16]. For the most part, these contributions focus on the energy-efficient (and, typically, time-constrained) offloading of (parts of) application programs from mobile devices to nearby FC servers through the exploitation of WiFi/Cellular-based communication technologies (see, for example, [17] and references therein for an updated overview of this topic).

Up to now, a few papers [18–21] are devoted, indeed, to the novel area of the integration of DNN models onto FC platforms. In this regard, the authors of [19] developed a proof of concept that aims at deploying DNNs onto Fog nodes for ML-based health prognosis. The underlying driving principle is to perform a suitable search among the candidate Fog nodes, in order to find free nodes with sufficient spare resources to delegate mining tasks. The work in [20] proposes an ecosystem that exploits the synergic cooperation of mobile devices and Fog nodes running DNN models for fast object recognition. In [21], a DNN-based face recognition application is simulated and the obtained performance exhibits reduced mining time when smartphones' photos are processed by proximate Fog nodes as opposed to the remote Cloud. The work in [18] generalizes this result, and gives some numerical evidence that the exploitation of Fog platforms for the joint fusion and distributed mining of sensor-acquired data may reduce the processing time and the energy-bandwidth consumption with respect to the corresponding fully centralized implementation of DNN models at the remote Cloud data centers. Overall, like our paper, these contributions test the feasibility of FC platforms for the distributed execution of DNN models under various operating conditions. However, *unlike* our paper, they do not formally address the problem of the optimized mapping of the computing tasks needed to support the DNN models for distributed Fog platforms.

3. The Proposed SmartFog Platform

The proposed SmartFog technological platform is sketched in Figure 1. It consists of N virtualized Fog nodes and a virtualized Cloud node that can communicate with each other and the SMs through a set of Network Interface Cards (NICs). Each Fog node can gather data measurements from several SMs by using a Multiplexer (MUX). It uses an SDAE [3–5] for data analytics and a networked Fog platform for SDAE pretraining and execution.

In this regard, we shortly point out that a Stacked Auto-Encoder (SAE) is a feed-forward neural network with $2L-1$ hidden layers trained to (quasi) *reproduce* its input at the output layer (see Figure 2). The aim of an SAE is to learn a compressed and distributed representation $\mathbf{h}_L$ (*encoding*) for a set of (possibly, noisy) input data $\mathbf{x}$ (typically for the purpose of dimensionality reduction), using a set of weight matrices $\mathbf{W}_k$, $k = 1, \ldots, L$. Then, an estimate $\widehat{\mathbf{x}}$ of the data $\mathbf{x}$ is recovered (*decoding*) by using tied weights (i.e., $\mathbf{W}_k^* = \mathbf{W}_k^T$), as depicted in Figure 2 [4]. As a robust variant, the Stacked *Denoising* Auto-Encoder (SDAE) [5] is a stochastic version of the SAE, where a stochastically corrupted $\tilde{\mathbf{x}}$ version of the input $\mathbf{x}$ is employed to feed the SAE (usually, using a Gaussian additive noise), while the uncorrupted input $\mathbf{x}$ is still used as the target for the optimization of the weight matrices (see Figure 2).

As shown in Figure 2, the computation of the k-th layer in the encoding phase is performed according to the following relationship [5]:

$$\mathbf{h}_k = \sigma_k \left(\mathbf{W}_k \mathbf{h}_{k-1} + \mathbf{b}_k \right), \quad k = 1, \ldots, L, \tag{1}$$

With $\mathbf{h}_0 = \tilde{\mathbf{x}}$, while the companion decoding layer acts as follows [5]:

$$\widehat{\mathbf{h}}_k = \mathbf{W}_{k+1}^T \widehat{\mathbf{h}}_{k+1} + \mathbf{c}_k, \quad k = L-1, \ldots, 0. \tag{2}$$

In Equations (1) and (2), $\mathbf{W}_k$ is the weight matrix of the k-th layer of the considered SDAE, $\mathbf{b}_k$ and $\mathbf{c}_k$ are the bias vectors of the encoder and decoder at the k-th layer, $\sigma(\cdot)_k$ is the element-wise nonlinear activation function of the k-th layer of the encoder (e.g., sigmoid or ReLU) and $\widehat{\mathbf{h}}_0 \equiv \widehat{\mathbf{x}}$. The decoder transformation in (2) is of affine type due to the continuous-valued nature of the input vector $\mathbf{x}$.

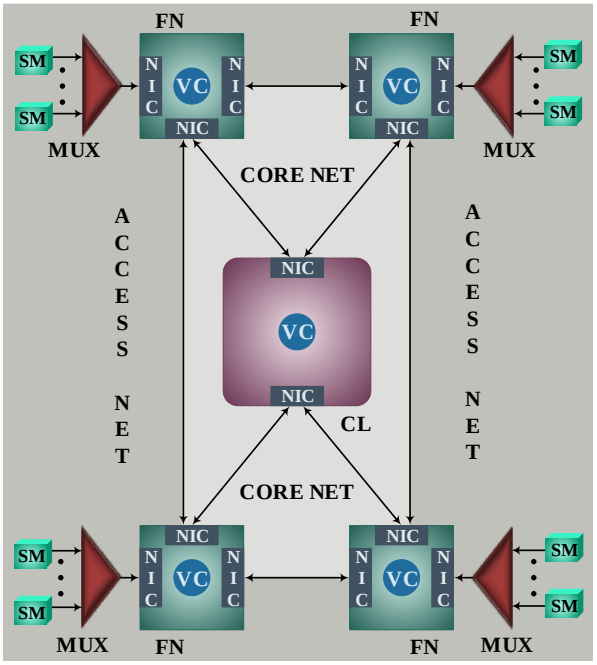

Figure 1. A sketch of the proposed SmartFog technological platform. Arrowed paths indicate data links. SM: = Smart-Meter; MUX: = Multiplexer; FN: = Fog Node; CL: = Cloud; VC: = Virtual Clone; NIC: = Network Interface Card.

The unsupervised pretraining of such an architecture is done on a per-layer basis [5]. Specifically, each layer is trained as a denoising auto-encoder by minimizing with respect to $\mathbf{h}_k$, $\mathbf{b}_k$ and $\mathbf{c}_k$ the total squared reconstruction error:

$$\mathcal{L}\left(\widehat{\mathbf{h}}_k\right) \triangleq \frac{1}{2} \sum_t \left\| \widehat{\mathbf{h}}_k^t - \mathbf{h}_k^t \right\|_2^2, \quad k = 1, \ldots, L, \tag{3}$$

where t is the batch index [5] and $\mathbf{h}_k^t$ is the t-th batch of the input vector at layer k. In our framework, the minimization of (3) is pursued by applying the Adam algorithm, a variant of the stochastic gradient descent that is based on the adaptive estimation of the first and second order moments of the gradient of (3) with respect to weights and bias of Figure 2 [4].

A k-NN classifier [4] is used atop the final hidden representation $\mathbf{h}_L$ to detect whether or not the input $\mathbf{x}$ contains an anomaly. The implemented classifier is binary and its output is 0 for "no anomalies" while it is 1 for "anomalies" detected in the data $\mathbf{x}$. The k-NN classifier assigns the input to the class with most examples among the k neighbors of the input. All neighbors have equal vote, and the class with the maximum number of votes among the k neighbors is selected. For this purpose, similarity is defined according to a suitable distance metric (usually the Euclidean one) between two data points.

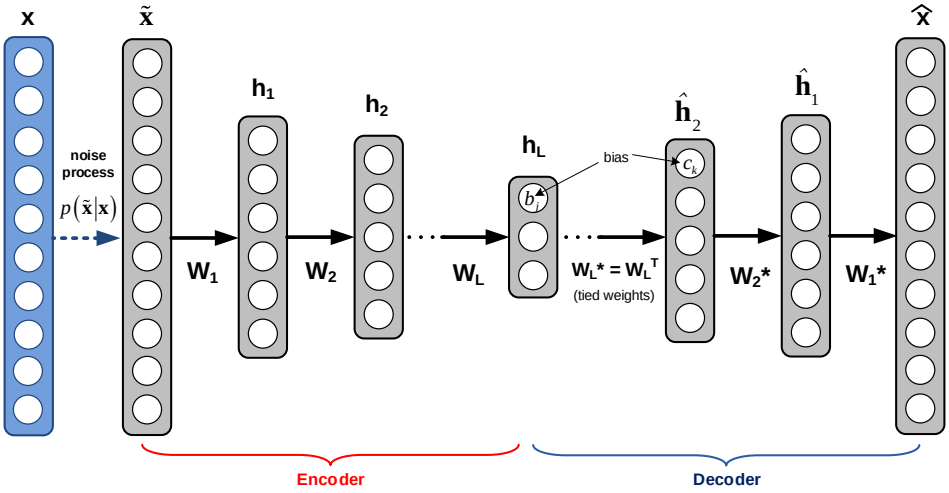

Figure 2. The structure of the considered SDAE. $\mathbf{x}$, $\tilde{\mathbf{x}}$ and $\hat{\mathbf{x}}$ are equal-size and real-valued vectors that represent the input, noise-corrupted and recovered data, respectively. $\mathbf{W}_k^* = \mathbf{W}_k^T$ is the k-th tied weight matrix.

4. Implementing SDAE over the Computing Nodes of SmartFog

To perform the per-layer pretraining over the available Fog and Cloud nodes of Figure 1, we propose a (novel) adaptive version of the elitist GA [9]. The goal is to minimize the total computing-plus-networking energy consumed by the resulting SmartFog platform of Figure 1. Specifically, in the implemented GA, the chromosome is the vector $\mathbf{a} \in \mathbb{R}^{L\times 1}$ containing the L indexes of the involved computing nodes. The goodness of each chromosome $\mathbf{a}$ is measured through an appropriate fitness function. In each generation, a set of elements is selected by comparing their fitness, the best element is retained, a fraction of the "best" elements of the current population is recombined through crossover, while the remaining "worst" elements are modified by randomly mutating each one over randomly selected positions. For this purpose, the implemented Crossover function generates a random pointer to the location index at which the crossover is performed and, then, carries out the corresponding swapping [4]. After generating a random vector of pointers to the locations to be mutated, the Mutation function returns the generated mutated values at the pointed locations of the vector $\mathbf{a}$. The fitness function used by SmartFog is the *inverse* of the total computing-plus-networking energy $\mathcal{E}$ consumed by the SmartFog platform for the execution of these operations. It equates to [7]:

$$\begin{aligned}\mathcal{E} = &\sum_{j=1}^{N} P_j^{(cmp)} \Delta t_j^{(cmp)} + \sum_{j=1}^{N} P_j^{(C-idle)} \Delta t_j^{(C-idle)} \\ &+ \sum_{j=1}^{N} P_j^{(net)} \Delta t_j^{(net)} + \sum_{j=1}^{N} P_j^{(N-idle)} \Delta t_j^{(N-idle)}.\end{aligned} \tag{4}$$

According to the clone-based virtualized Fog architecture detailed in [7], in (4), we have that: (i) $P_j^{(cmp)}$ is the power consumed by the virtual clone at the j-th Fog node for the computing; (ii) $P_j^{(C-idle)}$ is the power consumed by the j-th Fog node in the idle state; (iii) $P_j^{(net)}$ is the power consumed by the virtual clone at the j-th Fog node for the communication with the SM, the Cloud node and/or with other Fog nodes; and, (iv) $P_j^{(N-idle)}$ is the power consumed by the j-th network card in the idle state, while the related Δt_j is the execution time of state j. The power consumed by the computing nodes of Figure 1 for computing and networking is proportional to the squared values of the CPU frequencies and communication bandwidths [7]. Detailed analytical models for the evaluation of the computing/networking power present in (4) are developed in [7] and, for the sake of brevity, are not replicated here.

5. The Proposed Adaptive Elitist GA

A main advantage of the (previously mentioned) elitist mechanism is that it guarantees that, by design, the "best" chromosome delivered at the current iteration is also *globally* the best chromosome generated up to the current iteration [22]. This implies that, in our framework, the sequence of the energy returned by the Elitist GA is guaranteed to exhibit a non-increasing behavior. However, this pro is typically counterbalanced by one major disadvantage. In fact, elitism is, by design, a conservative strategy that tends to enhance the exploitation capability of the GA by reducing the corresponding exploration capability [22]. This means, in turn, that elitist GAs tend to be trapped by the local minima of the underlying objective functions and, then, they are slow to converge [22].

Motivated by these considerations, we propose a suitable *Adaptive* version of the standard Elitist GA (namely the AEGA), whose ultimate goal is to attain an improved trade-off between the two contrasting requirements of high exploitation and exploration capabilities.

Algorithm 1 reports a pseudo-code of the proposed AEGA, together with a description of its input, output and local variables. Roughly speaking, the following main design principles are on the basis of the proposed AEGA:

1. the size S_{POP} of the full population to be generated at each iteration is specified as an input parameter and it is held constant (see the list of the input parameters of Algorithm 1). However, the sizes S_{ELIT} and S_{CHA} of the sub-lists of the elite chromosomes and the crossed over-plus-mutated chromosomes are *adaptively* updated at each iteration. The goal is to adaptively balance the exploitation-vs.-exploration capabilities of the resulting AEGA (see the "for" statement at line #7 of Algorithm 1), while guaranteeing that the summation: $S_{ELIT} + S_{CHA}$ remains equal to S_{POP} (see lines #14 and #18 of Algorithm 1);
2. at each iteration, the relative (absolute) gap: $|\mathcal{E}_{CUR} - \mathcal{E}_{PRE}| / \mathcal{E}_{CUR}$ between the energy $\mathcal{E}_{CUR}$ and $\mathcal{E}_{PRE}$ returned by the AEGA at the current iteration and the previous one is used for indicating if the algorithm is trapped by the attraction basin of an *already* explored local minimum or is just entering the still unexplored attraction basin of the *new* local minimum (see the two "if" statements at lines #12 and #16 of Algorithm 1);
3. when the relative energy gap is below a (user defined) lower threshold TH_{LOW} (see line #12 of Algorithm 1), it is reasonable to expect that the algorithm is currently trapped by the attraction basin of an *already explored* local minimum. Hence, in order to quickly escape it, the exploration capability of the AEGA should be increased. For this purpose, the size S_{ELIT} of the list of the elite chromosomes is reduced by a (user defined) factor r, with $0 < r < 1$ (see line #13 of Algorithm 1), while the corresponding size S_{CHA} of the list of the crossed over-plus-mutated chromosomes is increased as reported in line #14 of Algorithm 1. Such updated values of S_{ELIT} and S_{CHA} will be used for performing the next iteration (see the "for" statement at line #7 of Algorithm 1);
4. when the relative energy gap is larger than a (user defined) upper threshold TH_{UPP} (see line #16 of Algorithm 1), it is reasonable to expect that the AEGA is *just* entering the *still unexplored* attraction basin of a new local/global minimum. Hence, in order to finely examine this new

region of the solution landscape, the exploitation capability of the AEGA should be increased. For this purpose, the size S_{CHA} of the list of the crossed over-plus-mutated chromosomes is reduced by the factor r (see line #17 of Algorithm 1), while the size S_{ELIT} of the list of the elite chromosomes is increased as reported in line #18 of Algorithm 1;

5. finally, when the relative energy gap falls into the inter-threshold interval $[TH_{LOW}, TH_{UPP}]$, S_{ELIT} and S_{CHA} remain unchanged.

Algorithm 1 Adaptive Elitist GA (AEGA).

Input: S_{POP}: integer-valued population size;
TH_{LOW}: positive-valued lower threshold;
TH_{UPP}: positive-valued upper threshold, with $TH_{LOW} \geq TH_{UPP}$;
I_{GEN}: number of performed iterations;
r: reducing factor, with $0 < r < 1$.

Output: $\mathcal{E}_{BEST}$: energy consumed by the globally best generated chromosome.

Local variables: $\mathcal{E}_{CUR}$: energy consumed by the best chromosome of the current population;
$\mathcal{E}_{PRE}$: energy consumed by the best chromosome of the previous population;
S_{ELIT}: integer-valued size of the current set of elite chromosomes;
S_{CHA}: integer-valued size of the current set of crossed over-plus-mutated chromosomes, with $S_{CHA} + S_{ELIT} = S_{POP}$;
$\mathcal{L}$: full list (of size S_{POP}) of the currently generated chromosomes.

— Initialization phase —

1: Randomly generate a list $\mathcal{L}$ of S_{POP} chromosomes of size $L + 1$;
2: Compute the energy of the generated chromosomes through Eq. (4);
3: Sort the chromosomes into ascending order of their consumed energy values and return the sorted list $\mathcal{L}$;
4: Store into $\mathcal{E}_{PRE}$ the energy of the first (i.e., the best) element of the sorted list$\mathcal{L}$;
5: Initialize the size S_{ELIT} of the set of the elite chromosomes, with $1 \leq S_{ELIT} \leq S_{POP}$;
6: Initialize the size S_{CHA} of the set of the crossed over-plus-mutated chromosomes at $S_{CHA} = S_{POP} - S_{ELIT}$;

— Iterative phase —

7: **for** $i = 1 : I_{GEN}$ **do**
8: Perform the crossover and mutation of the last S_{CHA} chromosomes of the current population and return the updated population list $\mathcal{L}$;
9: Evaluate the energy of the S_{POP} chromosomes of the returned list $\mathcal{L}$;
10: Sort the chromosomes into ascending order of their consumed energy values and return the sorted list $\mathcal{L}$;
11: Store the energy of the first chromosome of the sorted list $\mathcal{L}$ into $\mathcal{E}_{CUR}$;
12: **if** $(|\mathcal{E}_{CUR} - \mathcal{E}_{PRE}| / \mathcal{E}_{CUR}) < TH_{LOW}$ **then** ▷ Increase the *exploration* capability of the AEGA
13: $S_{ELIT} = \text{round}\,(rS_{ELIT})$;
14: $S_{CHA} = S_{POP} - S_{ELIT}$;
15: **end if**
16: **if** $(|\mathcal{E}_{CUR} - \mathcal{E}_{PRE}| / \mathcal{E}_{CUR}) > TH_{UPP}$ **then** ▷ Increase the *exploitation* capability of the AEGA
17: $S_{CHA} = \text{round}\,(rS_{CHA})$;
18: $S_{ELIT} = S_{POP} - S_{CHA}$;
19: **end if**
20: Update: $\mathcal{E}_{PRE} = \mathcal{E}_{CUR}$;
21: **end for**

— End of the iterative phase —

22: $\mathcal{E}_{BEST} = \mathcal{E}_{CUR}$; ▷ Store the energy of the globally best generated chromosome
23: **return** $\mathcal{E}_{BEST}$.

Due to the elitist nature of the proposed AEGA, the energy $\mathcal{E}_{CUR}$ returned at the end of the last iteration (i.e., at $i \equiv I_{GEN}$; see line #7 of Algorithm 1) is also the minimum energy obtained over the overall set of carried out iterations (see line #22 of Algorithm 1). Hence, it constitutes the final output of the AEGA (see line #23 of Algorithm 1).

Before proceeding, three main remarks about the expected impact on the performance of the AEGA of the setting of the input parameters TH_{LOW}, TH_{UPP} and r of Algorithm 1 are in order. First, since these parameters play, indeed, the role of *hyper*-parameters, their optimized setting depends on the actual landscape of the adopted objective function, and, then, we must resort to numerical trials for their optimization (see the following Section 6 for additional details on this topic). Second, we expect that, by design, the exploration capability of the proposed AEGA increases for increasing values of the lower threshold TH_{LOW} (see line #12 of Algorithm 1), while lower values of the upper threshold TH_{UPP} enhances the corresponding exploitation capability (see line #16 of Algorithm 1). Third, we expect that the sensitivity of the AEGA on local variations of the landscape of the adopted objective function increases for decreasing values of the reducing factor r. This means, in turn, that lower values of r may shorten the transient phase and speed up the convergence of the AEGE toward *good* local (or even global) minima, but, at the same time, they may also induce oscillation phenomena in the converged performance. Therefore, we expect that the right setting of r is the suitable trade-off among the two contrasting requirements of short transient phases and stable performance at the convergence.

6. Numerical Performance of the Proposed SmartFog

The goal of this section is threefold. First, we describe the implemented setting for the simulation of the proposed SmartFog platform. Second, we numerically test the accuracy performance of the proposed SDAE of Figure 2 and compare it with the corresponding one of a benchmark solution that uses a shallow Support Vector Machine (SVM) for anomaly detection. Third, we test the energy performance of the SmartFog platform equipped with the proposed AEGA of Algorithm 1 and compare it to the corresponding ones of two benchmark solutions. Specifically, the first benchmark solution performs the mapping of the SDAE layers of Figure 2 onto the SmartFog platform of Figure 1 by resorting to a *non-adaptive* conventional Elitist GA, while the second one uses only the Cloud node for performing the training of the implemented SDAE of Figure 1.

Implemented test framework—Numerical tests of the SmartFog platform have been carried out by using the UCI "Individual household electric power consumption" data set available at: https://archive.ics.uci.edu/ml/datasets/individual+household+electric+power+consumption. It embraces more than 2 millions of measurements gathered in a house located in Sceaux (near Paris, France) between December 2006 and November 2010. Instances in the dataset represent the active per-minute energy (in Watt hour) consumed by the household and measured by the SM. From this data set, we have selected 30 days from 10 SMs (43,200 instances) for the train set and 10 days (14,400 instances) for the test set. The number of anomalies in the data set is about 12% of the total. These anomalies consist of time-series representing faulty meters or fraudulent users.

The performance of the proposed SmartFog platform was simulated by leveraging the recently proposed VirtFogSim toolbox [9]. It provides several primitives for the customized simulations of virtualized (i.e., clone-based) networked Fog ecosystems.

The input provided to the simulated SDAE consists of frames of duration of one, two, four and six hours, which are equivalent to 60, 120, 240 and 360 input units, respectively. A Gaussian noise with zero mean and a variance equal to 1% of the mean squared value of the considered training data was added to the input vector $\mathbf{x}$, in order to guarantee a robust classification (see Figure 2).

We tested two SDAEs with $L = 3$ (3L-SDAE) and $L = 5$ (5L-SDAE) hidden layers and four different lengths of the input vector $\mathbf{x}$. The number of units in the hidden layers are detailed in Table 1. We used the sigmoid function as nonlinear function in all layers. In the implemented Adam algorithm [4], the learning rate is set to $\mu = 0.001$ for all layers, and the hyper-parameters β_1 and β_2 [4]

are set to $\beta_1 = 0.9$ and $\beta_2 = 0.999$, respectively. For the k-NN classifier, we have used 5 neighbors and the Euclidean distance as similarity measure.

Table 1. Per-layer hidden units in the tested 3L-SDAE and 5L-SDAE architectures.

Input L.	First L.	Second L.	Third L.	Fourth L.	Fifth L.
3L-SDAE—3 Hidden Layers					
60	30	20	10	—	—
120	60	40	20	—	—
240	120	80	40	—	—
360	180	120	60	—	—
5L-SDAE—5 Hidden Layers					
60	40	30	20	10	5
120	80	60	40	20	10
240	160	120	80	40	20
360	240	180	120	60	30

Comparative accuracy performance—The obtained numerical results in terms of classification accuracy are reported in Table 2. Accuracy is defined as the ratio between the number of instances correctly classified out the total instances. From Table 2, we can argue that the accuracy decreases when the length of the analysis window is too short or too long: interestingly, a time window of two hours produces the best accuracy. Moreover, we can see that the use of three hidden layers produces better results than five hidden layers. Hence, the 3L-SDAE with a time window of two hours can be considered as the most reliable solution for detecting anomalies in SM data.

Table 2. Tested classification accuracy of the 3L-SDAE and 5L-SDAE architectures.

Time Window [h]	# Inputs	Accuracy 3L-SDAE	Accuracy 5L-SDAE
1	60	0.854	0.792
2	120	0.926	0.884
4	240	0.869	0.801
6	360	0.795	0.725

As a benchmark, a Support Vector Machine (SVM) classifier was trained on the same set of raw data. The corresponding soft-margin parameter C was optimized by a grid search algorithm over the set: $C \in \{10^{-3}, 10^{-2}, \ldots, 10^{6}, 10^{7}\}$, selecting the best value of $C = 100$. The accuracy obtained by the tested SVM is detailed in Table 3. This table puts in evidence that the SVM produces a worse accuracy than the 3L-SDAE architecture of about 5%.

Table 3. Classification accuracy for the benchmark SVM classifier.

Time Window [h]	Accuracy
1	0.826
2	0.885
4	0.811
6	0.808

Comparative energy performance— Passing to consider the energy consumption of the underlying SmartFog execution platform of Figure 1, we carried out several tests by varying the strategy for the mapping of the hidden layers of SDAE of Figure 2 onto the available Fog-Cloud computing nodes of Figure 1. For this purpose, three mapping strategies were implemented and numerically tested. The first one relies on the proposed AEGA of Algorithm 1. The second one implements a *Non-Adaptive* Elitist GA (NA-EGA), that is obtained by removing from the pseudo-code of Algorithm 1 both the "if"

statements at lines #12 and #16. The last strategy is the simplest one and maps all the hidden layers of the considered SDAE of Figure 2 to the Cloud node of Figure 1 for their training. In this last case, no energy consumption can be obtained since the entire algorithm is running on the unique Cloud node. In all performed tests, the size S_{POP} of the populations of the simulated GAs is held constant, equal to 10, the only-Cloud solution is included in the initial populations and a maximum of $I_{GEN} = 30$ iterations are run. Furthermore, the (numerically optimized) hyper-parameters TH_{LOW}, TH_{UPP} and r used for the simulation of the proposed AEGA of Algorithm 1 are set to 0.1, 1.1 and 0.85 respectively, while, in the simulated NA-EGA, the (numerically optimized) sizes S_{ELIT} and S_{CHA} of the elite and crossed over-plus-mutated sub-lists are fixed to 4 and 6, respectively. In the following, the optimized inter-threshold interval will be denoted as $\mathcal{T}_0$.

According to Figure 1, the simulated SmartFog platform is composed of five Fog nodes and a single Cloud node, with each node being equipped with a corresponding virtual clone [9]. The computing and network capacities of each Fog clone are randomly selected over the sets: 5–12 Mbit/s for the processing rates of CPUs, 5–10 Mbit/s for the SM-to-Fog transmission rates and 10–20 Mbit/s for the intra-Fog and the Fog-Cloud transmission rates. However, the computing and network capacities of the Cloud clone are fixed to their respective maximum values (i.e., 12, 10 and 20 Mbit/s, respectively), in order to reflect the more stable and larger resources made available by the Cloud. Furthermore, the simulated inter-node network topology is meshed, with inter-Fog and Fog-Cloud links that are bidirectional and symmetric. Since we consider here the 3L-SDAE, we need to allocate $L = 3$ hidden layers over the available computing (i.e., Fog-plus-Cloud) nodes.

The resulting energy plots are reported in Figure 3. Each simulated point was obtained by averaging over 30 independent runs of randomly generated computing and network capacities of the (aforementioned) five Fog nodes that compose the simulated SmartFog platform. Specifically, in Figure 3, the continuous black (resp., dashed blue) curve plots the simulated average energy consumption of the proposed AEGA of Algorithm 1 under the optimized inter-threshold $\mathcal{T}_0 = [0.1, 1.1]$ (resp., the benchmark NA-EGA), while the point marked by the red diamond reports the average energy consumption of the (aforementioned) only-Cloud solution that uses only the Cloud node for the training of the implemented 3L-SDAE.

An examination of the energy plots of Figure 3 leads to three main insights.

First, being of elitist type, both the proposed AEGA and benchmark NA-EGA converge to the same final energy value by following non-increasing trajectories. However, the convergence time (in multiple of the generated populations) of the proposed AEGA (resp., benchmark NA-EGA) is around 16 (resp., 27), so that the convergence of the AEGA is about 1.7 times faster than the corresponding one of the benchmark NA-EGA.

Second, the behavior of the energy curve of the AEGA is step-like (see the cliffs at the iteration indexes 3, 6 and 16 in Figure 3), while the corresponding behavior of the NA-EGA is smoother. We have numerically ascertained that the step-like behavior of the energy plot of the AEGA is induced, indeed, by the adaptive mechanism implemented by the "if" statements at lines #12 and #16 of Algorithm 1, which allows the AEGA to quickly escape already explored regions of the underlying objective function and to fast enter still unexplored attraction basins of new local/global minima.

Third, the gap between the energy consumption of the only-Cloud solution (i.e., the red diamond in Figure 3) and the steady-state one of the proposed AEGA is noticeable (e.g., around 30%) and it is approached by the AEGA after several iterations limited up to 16 (see the continuous curve of Figure 3).

Sensitivity on the threshold setting—In this paragraph, we consider the impact of the setting of the two hyper-parameters TH_{LOW} and TH_{UPP} on the average energy consumption of the proposed SmartFog platform. Specifically, we performed the tests by using the following two additional inter-threshold intervals $\mathcal{T}_1 = [0.3, 0.6]$ and $\mathcal{T}_2 = [0.05, 2.0]$, respectively, while the hyper-parameter r was set to the previous value of 0.85. The average energy consumption of the proposed AEGA of Algorithm 1 under the inter-threshold intervals $\mathcal{T}_1$ and $\mathcal{T}_2$ is shown by the dotted dark green curve and the dash-dotted magenta curve of Figure 3, respectively.

Interestingly enough, we can observe from the dotted dark green curve of Figure 3 that when the more stressed inter-threshold interval $\mathcal{T}_1$ is used, the proposed AEGA converges to the same final energy value of the optimized interval $\mathcal{T}_0$. However, the convergence time is very different from the previous one. Since, in fact, the threshold are more restrictive, the proposed AEGA shows a fast transitory behavior, but after a few iterations the convergence becomes flat (the thresholds are too restrictive and more iterations are needed) and the time period for obtaining the final solution is stretched. The total convergence time is around 20 iterations. On the other hand, when the more relaxed inter-threshold interval $\mathcal{T}_2$ is used, the dash-dotted magenta curve of Figure 3 shows that the convergence behavior tends to emulate the shape of the non-adaptive GA strategy depicted by the dashed blue curve of Figure 3. In this case, the convergence time is about 24 iterations while the AEGA converges to the same final energy value of the optimized interval $\mathcal{T}_0$.

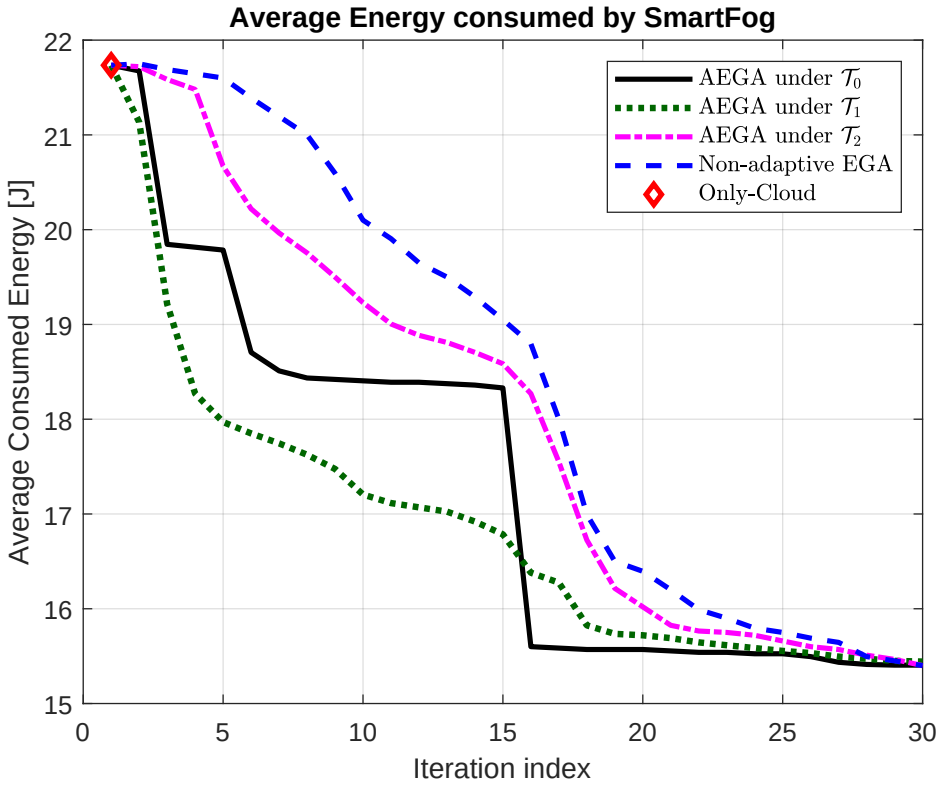

Figure 3. Average energy consumption of the simulated SmartFog platform of Figure 1 for the training of the 3L-SDAE under the: AEGA with the optimized inter-threshold $\mathcal{T}_0$ (continuous black curve), AEGA with the inter-threshold $\mathcal{T}_1$ (dotted dark green curve), AEGA with the inter-threshold $\mathcal{T}_2$ (dash-dotted magenta curve), NA-EGA (dashed blue curve) and only-Cloud (red diamond) mapping solutions.

7. Conclusions and Hints for Future Research

In this contribution, we focused on the optimized design of SmartFog, a Fog-based networked computing platform for the distributed training of deep auto-encoders for anomaly detection in data streams generated by Smart-Meters (SMs). For this purpose, after defining the main building blocks of the SmartFog platform, we proposed a new *Adaptive* Elitist GA (i.e., the AEGA) for the energy-efficient distributed training of the implemented SDAE. The performance of the resulting optimized SmartFog platform was numerically tested and compared with the corresponding ones of some state-of-the-art benchmark solutions in terms of accuracy, energy consumption and execution time.

Overall, the main insights stemming from the reported numerical results are that: (i) the proposed SmartFog platform reaches an accuracy of 92.6% by using an SDAE with three hidden layers and a

time window of two hours; and, (ii) the proposed AEGA equipping the SmartFog platform allows us to reduce the convergence time of about 1.6 times with respect to the conventional NA-GA, while guaranteeing a 30% of energy saving with respect to the only-Cloud benchmark solution.

Being the integration of the Deep Neural Networks and Fog Computing paradigms in its infancy [10], we believe that the reported results are, indeed, only the tip of the iceberg and, then, they could be further developed along (at least) three main research lines.

First, the present paper focuses on the energy-efficient implementation of SDAEs atop Fog platforms. However, SDAEs represent only an instance of the overall family of DNNs [4]. Hence, the generalization of the presented results to DNN models described by general directed acyclic graphs could be a first research line of potential interest.

Second, this work focuses on the training phase of the SDAEs, that is typically considered the most cumbersome one from a computational point of view. However, emerging IoT-oriented real-time applications demand for stringent delay requirements during the inference phase [10]. Hence, the design and testing of distributed Fog technological platforms for the energy-saving and real-time inference of big data through DNNs may be a second research line of potential interest.

Finally, the presented results rely on the (implicit) assumption that both the volume of data to be processed and the states (that is, the bandwidths) of the communication links of the SmartFog platform of Figure 1 stay unchanged over time intervals at least equal to the time required to carry out the training of the SDAE of Figure 2. As (recently) pointed out, for example, in [23], this assumption is typically met in SmartGrid networks for emerging Green Smart Home applications, in which both SMs and Fog nodes are statically installed in buildings that communicate over fixed power line-based wired links (see, for example, Section 3 of [23] and the reference framework of Figure 2). However, in principle, both the here proposed SDAE-based approach to the anomaly detection and the related AEGA-based resource management solution could be effectively employed even in the emerging field of the so-called Vehicular Fog Computing (VFC) [24,25]. In this environment, devices on board of parked or slowly moving vehicles are used to form a wireless (possibly, mobile) intermediary Fog layer. These devices collect the data from wireless sensors installed on the vehicles, mine them and communicates the pre-processed results to remote Clouds for further processing by using nearby Road Side Units (see, for example, Figures 1–4 of [24] for some sketches of the underlying reference scenarios). Hence, in principle, VFC fits the general architecture in Figure 1 of the proposed SmartFog technological platform provided that (see Figure 1): (i) SMs are replaced by wireless (possibly, mobile) sensors; (ii) Fog nodes are replaced by vehicles; and, (iii) both inter-Fog and Cloud-Fog links are assumed to be wireless. However, in the resulting VFC scenario, both the assumptions of time-invariant volume of data to be processed and time-invariant link bandwidths may fall short [26,27]. This needs the introduction of imperfect channel estimation techniques [28–30]. How the proposed SmartFog platform could be refined for effectively coping with the environmental time fluctuations of VFC-based application scenarios could be an additional hint for future research.

Author Contributions: M.S., E.B., A.M. and A.U. contributed equally to this work.

Funding: This work was supported by the project: "GAUChO—A Green Adaptive Fog Computing and networking Architectures" funded by the MIUR Progetti di Ricerca di Rilevante Interesse Nazionale (PRIN) Bando 2015, Grant 2015YPXH4W_004, and by the projects: "Vehicular Fog: energy-efficient QoS mining and dissemination of multimedia Big Data streams" (V-Fog and V-Fog2), and: "SoFT: Fog of Social IoT", funded by Sapienza University of Rome, Bandi 2016, 2017 and 2018.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AE	Auto-Encoder
AEGA	Adaptive Elitist Genetic Algorithm
CC	Cloud Computing
DBN	Deep Belief Networks
DNN	Deep Neural Network
FC	Fog Computing
FN	Fog Node
GA	Genetic Algorithm
MUX	Multiplexer
NA-EGA	Non-Adaptive Elitist Genetic Algorithm
NIC	Network Interface Card
NN	Neural Network
SAE	Stacked Auto-Encoder
SDAE	Stacked Denoising Auto-Encoder
SM	Smart-Meter
SVM	Support Vector Machine
VC	Virtual Clone
VFC	Vehicular Fog Computing

References

1. Rossi, B.; Chren, S.; Buhnova, B.; Pitner, T. Anomaly detection in Smart Grid data: An experience report. In Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC 2016), Budapest, Hungary, 9–12 October 2016; pp. 2313–2318, doi:10.1109/SMC.2016.7844583.
2. Yip, S.C.; Tan, W.N.; Tan, C.; Gan, M.T.; Wong, K. An anomaly detection framework for identifying energy theft and defective meters in smart grids. *Int. J. Electr. Power Energy Syst.* **2018**, *101*, 189–203, doi:10.1016/j.ijepes.2018.03.025.
3. Pereira, J.; Silveira, M. Unsupervised Anomaly Detection in Energy Time Series Data Using Variational Recurrent Autoencoders with Attention. In Proceedings of the 17th IEEE International Conference on Machine Learning and Applications (ICMLA 2018), Orlando, FL, USA, 17–20 December 2018; pp. 1275–1282, doi:10.1109/ICMLA.2018.00207.
4. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
5. Vincent, P.; Larochelle, H.; Lajoie, I.; Bengio, Y.; Manzagol, P.A. Stacked Denoising Autoencoders: Learning Useful Representationsina Deep Network with a Local Denoising Criterion. *J. Mach. Learn. Res.* **2010**, *11*, 3371–3408.
6. Baccarelli, E.; Biagi, M.; Bruno, R.; Conti, M.; Gregori, E. Broadband Wireless Access Networks: A Roadmap on Emerging Trends and Standards. In *Broadband Services: Business Models and Technologies for Community Networks*; Wiley Online Library: Hoboken, NJ, USA, 2005; Chapter 14, pp. 215–240, doi:10.1002/0470022515.ch14.
7. Baccarelli, E.; Vinueza Naranjo, P.G.; Scarpiniti, M.; Shojafar, M.; Abawajy, J.H. Fog of Everything: Energy-efficient Networked Computing Architectures, Research Challenges, and a Case Study. *IEEE Access* **2017**, *5*, 9882–9910, doi:10.1109/ACCESS.2017.2702013.
8. Shojafar, M.; Pooranian, Z.; P.G.Vinueza, P.G.; Baccarelli, E. FLAPS: Bandwidth and delay efficient distributed data searching in Fog-supported P2P content delivery networks. *J. Supercomput.* **2017**, *73*, 5239–5260, doi:10.1007/s11227-017-2082-y.
9. Scarpiniti, M.; Baccarelli, E.; Momenzadeh, A. VirtFogSim: A Parallel Toolbox for Dynamic Energy-Delay Performance Testing and Optimization of 5G Mobile-Fog-Cloud Virtualized Platforms. *Appl. Sci.* **2019**, *9*, 1160, doi:10.3390/app9061160.

10. Mohammadi, M.; Al-Fuqada, A.; Sorour, S.; Guizani, M. Deep learning for IoT Big data and Streaming analytics: A survey. *IEEE Commun. Surv. Tutor.* **2018**, *20*, 2923–2960, doi:10.1109/COMST.2018.2844341.
11. Reich, C.; Nicolaou, C.; Mansour, A.; Van Laerhoven, K. Bayesian Estimation of Recurrent Changepoints for Signal Segmentation and Anomaly Detection. In Proceedings of the 27th European Signal Processing Conference (EUSIPCO 2019), A Coruña, Spain, 2–6 September 2019; pp. 1–5.
12. Ozay, M.; Esnaola, I.; Vural, F.T.Y.; Kulkarni, S.R.; Poor, H.V. Machine learning methods for attack detection in the smart grid. *IEEE Trans. Neural Netw. Learn. Syst.* **2016**, *27*, 1773–1786, doi:10.1109/TNNLS.2015.2404803.
13. Wei, J.; Mendis, G.J. A deep learning-based cyber-physical strategy to mitigate false data injection attack in smart grids. In Proceedings of the Joint Workshop on Cyber-Physical Security and Resilience in Smart Grids (CPSR-SG), Vienna, Austria, 12 April 2016; pp. 1–6, doi:10.1109/CPSRSG.2016.7684102.
14. He, Y.; Mendis, G.J.; Wei, J. Real-Time Detection of False Data Injection Attacks in Smart Grid: A deep Learning-Based Intelligent Mechanism. *IEEE Trans. Smart Grid* **2017**, *8*, 2505–2516, doi:10.1109/TSG.2017.2703842.
15. Khan, A.U.R.; Othman, M.; Madani, S.A.; Khan, S.U. A survey of mobile cloud computing application models. *IEEE Commun. Surv. Tutor.* **2014**, *16*, 393–413, doi:10.1109/SURV.2013.062613.00160.
16. Mach, P.; Becvar, Z. Mobile edge computing. A survey on architecture and computation offloading. *IEEE Commun. Surv. Tutor.* **2017**, *19*, 1628–1656, doi:10.1109/COMST.2017.2682318.
17. Baccarelli, E.; Scarpiniti, M.; Momenzadeh, A. EcoMobiFog—Design and dynamic optimization of a 5G Mobile-Fog-Cloud multi-tier ecosystem for the real-time distributed execution of stream applications. *IEEE Access* **2019**, *7*, 55565–55608, doi:10.1109/ACCESS.2019.2913564.
18. Teerapittayanon, S.; McDanel, B.; Kung, H. Distributed deep neural networks over the cloud, the edge and end devices. In Proceedings of the IEEE 37th International Conference on Distributed Computing Systems (ICDCS 2017), Atlanta, GA, USA, 5–8 June 2017; pp. 328–339, doi:10.1109/ICDCS.2017.226.
19. Qaisar, S.B.; Usman, M. Fog networking for machine health prognosis: A deep learning perspective. In Proceedings of the International Conference on Computational Science and Its Applications (ICCSA 2017), Trieste, Italy, 3–6 July 2017; pp. 212–219, doi:10.1007/978-3-319-62404-4_16.
20. Li, D.; Salonidis, T.; Desai, N.V.; Chuah, M.C. DeepCham: Collaborative edge-mediated adaptive deep learning for mobile object recognition. In Proceedings of the 2016 IEEE/ACM Symposium on Edge Computing (SEC 2016), Washington, DC, USA, 27–28 October 2016; pp. 64–76, doi:10.1109/SEC.2016.38.
21. Yi, S.; Hao, Z.; Qin, Z.; Li, Q. Fog Computing: Platform and Applications. In Proceedings of the Third IEEE Workshop on Hot Topics in Web Systems and Technologies (HotWeb 2015), Washington, DC, USA, 12–13 November 2015; pp. 73–78, doi:10.1109/HotWeb.2015.22.
22. Talbi, E.G. *Metaheuristics: From Design to Implementation*; Wiley: Hoboken, NJ, USA, 2009.
23. Zahoor, S.; Javaid, S.; Javaid, N.; Ashraf, M.; Ishmanov, F.; Afzal, M.K. Cloud-Fog-Based Smart Grid Model for Efficient Resource Management. *Sustainability* **2018**, *10*, 2079, doi:10.3390/su10062079.
24. Hou, X.; Li, Y.; Chen, M.; Wu, D.; Jin, D.; Chen, S. Vehicular Fog Computing: A Viewpoint of Vehicles as the Infrastructures. *IEEE Trans. Veh. Technol.* **2016**, *65*, 3860–3873, doi:10.1109/TVT.2016.2532863.
25. Menon, V.G.; Prathap, J. Vehicular Fog Computing: Challenges Applications and Future Directions. In *Fog Computing: Breakthroughs in Research and Practice*; IGI Global: Hershey, PA, USA, 2018; Chapter 11, pp. 220–229, doi:10.4018/978-1-5225-5649-7.ch011.
26. Baccarelli, E.; Cusani, R.; Galli, S. A novel adaptive Equalizer wlith enhanced channel tracking capability for TDMA-based mobile radio communications. *IEEE J. Sel. Areas Commun.* **1998**, *16*, 1630–1639, doi:10.1109/49.737632.
27. Baccarelli, E.; Cusani, R. Recursive Kalman-type optimal estimation and detection of hidden Markov chains. *Signal Process.* **1996**, *51*, 55–64, doi:10.1016/0165-1684(96)00030-8.
28. Baccarelli, E.; Biagi, M. Power-allocation policy and optimized design of multiple-antenna systems with imperfect channel estimation. *IEEE Trans. Veh. Technol.* **2004**, *53*, 136–145, doi:10.1109/TVT.2003.822025.

29. Baccarelli, E.; Biagi, M. Performance and optimized design of space-time codes for MIMO wireless systems with imperfect channel estimates. *IEEE Trans. Signal Process.* **2004**, *52*, 2911–2923, doi:10.1109/TSP.2004.834269.
30. Baccarelli, E.; Biagi, M.; Pelizzoni, C. On the information throughput and optimized power allocation for MIMO wireless systems with imperfect channel estimation. *IEEE Trans. Signal Process.* **2005**, *53*, 2335–2347, doi:10.1109/TSP.2005.849165.