Energy and Buildings 116 (2016) 646–655

Contents lists available at ScienceDirect

Energy and Buildings

journal homepage: www.elsevier.com/locate/enbuild

Unsupervised energy prediction in a Smart Grid context using reinforcement cross-building transfer learning

Elena Mocanu*, Phuong H. Nguyen, Wil L. Kling[1], Madeleine Gibescu

Department of Electrical Engineering, Eindhoven University of Technology, 5600 MB Eindhoven, The Netherlands

ARTICLE INFO

Article history:
Received 15 June 2015
Received in revised form
11 December 2015
Accepted 22 January 2016
Available online 16 February 2016

Keywords:
Building energy prediction
Reinforcement learning
Transfer learning
Deep Belief Networks
Machine learning

ABSTRACT

In a future Smart Grid context, increasing challenges in managing the stochastic local energy supply and demand are expected. This increased the need of more accurate energy prediction methods in order to support further complex decision-making processes. Although many methods aiming to predict the energy consumption exist, all these require labelled data, such as historical or simulated data. Still, such datasets are not always available under the emerging Smart Grid transition and complex people behaviour. Our approach goes beyond the state-of-the-art energy prediction methods in that it does not require labelled data. Firstly, two reinforcement learning algorithms are investigated in order to model the building energy consumption. Secondly, as a main theoretical contribution, a Deep Belief Network (DBN) is incorporated into each of these algorithms, making them suitable for continuous states. Thirdly, the proposed methods yield a cross-building transfer that can target new behaviour of existing buildings (due to changes in their structure or installations), as well as completely new types of buildings. The methods are developed in the MATLAB® environment and tested on a real database recorded over seven years, with hourly resolution. Experimental results demonstrate that the energy prediction accuracy in terms of RMSE has been significantly improved in 91.42% of the cases after using a DBN for automatically extracting high-level features from the unlabelled data, compared to the equivalent methods without the DBN pre-processing.

1. Introduction

Prediction of energy consumption as a function of time plays an essential role in the current transition to future energy systems. Within the new context of so-called Smart Grids, the energy consumption of buildings can be regarded as a nonlinear time series, depending on many complex factors. The variability introduced by the growing penetration of wind and solar generation sources only strengthens the role of accurate prediction methods [1]. Prediction forms an integral part in the efficient planning and operation of the whole Smart Grid.

On the one hand, advanced energy prediction methods should be easily expandable to various levels of data aggregation at all time scales [2]. On the other hand, they have to automatically adapt with decision strategies based on dynamic behavior of active consumers (e.g. new and smart(er) buildings) [3]. Applications of these new methods should facilitate the transition from the traditional single-tariff grid to time-of-use (TOU) and real-time pricing. The effects will be felt by all players in the grid from transmission (TSO) and distribution system operator (DSO) to the end-user, including resource assessment and analysis of energy efficiency improvements, flexible demand response (DR) and other continuous projection on planning studies. The joint consideration of decisions regarding new renewable generation, TSO development, and demand-side management (DMS) programs in an integrated fashion requires demand forecasts. Consequently, these will require changes in the way how the data are collected and analyzed [4].

Prior studies have shown that by using statistical methods, more recent inspired by supervised machine learning techniques, such as Support Vector Machines [5,6], Artificial Neural Networks [7,8], Autoregressive models [9], Conditional random field [10], or Hidden Markov Model [11], one can improve the accuracy of energy prediction significantly. On the other hand, there are many methods based on physical principles, including a large number of building parameters, to calculate thermal dynamics and energy behavior at the building level. Moreover, to shape the evolution of future buildings systems, there are also some hybrid approaches which combine some of the above models to optimize predictive performance, such as [12–16]. Interested readers are referred for a more

* Corresponding author.
E-mail address: e.mocanu@tue.nl (E. Mocanu).
[1] Deceased March 14, 2015.

http://dx.doi.org/10.1016/j.enbuild.2016.01.030
0378-7788/

Nomenclature

α	learning rate, $\forall\alpha \in [0, 1]$
γ	discount factor, $\forall\gamma \in (0, 1)$
$\mathbb{E}[\cdot]$	expected value operator
$\mathcal{A}$	the set of actions, $\forall a \in \mathcal{A}$
$\mathcal{D}$	dataset
$\mathcal{R}$	the reward function
$\mathcal{S}$	the set of states, $\forall s \in \mathcal{S}$
$\mathcal{T}$	transition probability matrix
$\mathbf{h}$	vector collecting all the hidden units, $h_j \in \{0, 1\}$
$\mathbf{v}$	vector collecting all visible units, $v_i \in \mathbb{R}$
$\mathbf{W}^{vh}$	matrix of all weights connecting $\mathbf{v}$ and $\mathbf{h}$
E	total energy function in the RBM model
k	the number of hidden layers
M	building energy consumption model
p, P	probability value/vector
Q	the quality matrix
t	time
Z	normalization function

comprehensive discussion on the application of energy demand management to [9,14,17,18].

Although they remain at the forefront of academic and applied research, all these methods require labeled data able to faithfully reproduce the energy consumption of buildings. In the remaining of this paper we refer to the labeled data as to the historical (known) data of the analyzed building. Usually the lack of historical data can be replaced by simulated data. Still, both, historical or simulated data, are employed in these forecasting methods in a non-adaptable way without considering the future events or changes which can occur in the Smart Grid.

A stronger motivation for this paper, is given by the not too well exploited fact that sometimes there are not historically data consumption available for a particular building. From the machine learning perspective this is a typical unsupervised learning problem. One of the most used methods of unsupervised learning, reinforcement learning (RL), was introduced in power system area to solve stochastic optimal control problems [19]. RL methods are used in a wide range of applications, such as system control [20], playing games or more recent in transfer learning [19,21]. The advantage of the combination of reinforcement learning and transfer learning approaches is straightforward. Hence, we want to transfer knowledge from a global to a local perspective, to encode the uncertainty of the building energy demand.

Owing to the curse of dimensionality, these methods fail in high dimensions. More recently, there has been a revival of interest in combining Deep Learning with reinforcement learning. Therein, Restricted Boltzmann Machines were proven to provide a value function estimation [22] or a policy estimation [23]. More than that, Minh et al. [24] combined successfully deep neural networks and Q-learning to create a deep Q-network which successfully learned control policies in a range of different environments.

In this paper, we comprehensively explore and extended two reinforcement learning (RL) methods to predict the energy consumption at the building level using unlabelled historical data, namely State-Action-Reward-State-Action (SARSA) [25] and Q-learning [26]. Due to the fact that in the original form both methods can not handle well continuous states space, this paper contributes theoretically to extend them by incorporating a Deep Belief Network [27] for continuous states estimation and automatically features extraction in a unified framework. Our proposed RL methods are appropriate when we do not have historical or simulated data, but we want to estimate the impact of changes in Smart Grid, such as the appearance of a building or several buildings in a certain area, or more commonly a change in energy consumption due to building rennovation. In this paper, we have shown the applicability and efficiency of our proposed method in three different situations:

1. In the case of a new type of building being connected with the Smart Grid, thus transferring knowledge from a commercial building to a residential building. Specifically, in Section 6.2.1, four different types of residential buildings were analyzed.
2. In the case of a rennovated building, thus transferring knowledge from a non-electric heat building to a building with electric heating.
3. Additionally, we propose experiments to highlight the importance of external factors for the estimation of building energy consumption, such as price information. In Section 6.2.2, transfer learning is applied, from a building under a static tariff to a building with a time-of-use tariff.

According to our knowledge, this is the first time when the energy prediction is performed without using any information about that building, such as historical data, energy price, physical parameters of the building, meteorological condition or information about the user behavior.

The paper is organized as follows. In Section 2 we explain the rationality underlying our approach. Section 3 presents the mathematical modeling of the reinforcement learning approaches. Section 4 describes the novelty method to estimate continuous states in reinforcement learning using Deep Belief Networks. The experiments setup and results are illustrated in Sections 5 and 6, respectively. The paper concludes with a discussion and future work.

2. Problem formulation

In this paper, we propose a method to solve the unsupervised energy prediction problem with cross-building transfer by using machine learning time series prediction techniques. In the most general statement, the proposed Reinforcement and transfer learning setup is depicted in Fig. 1. Given the unevenly distributed building energy values during time, firstly, a special attention is

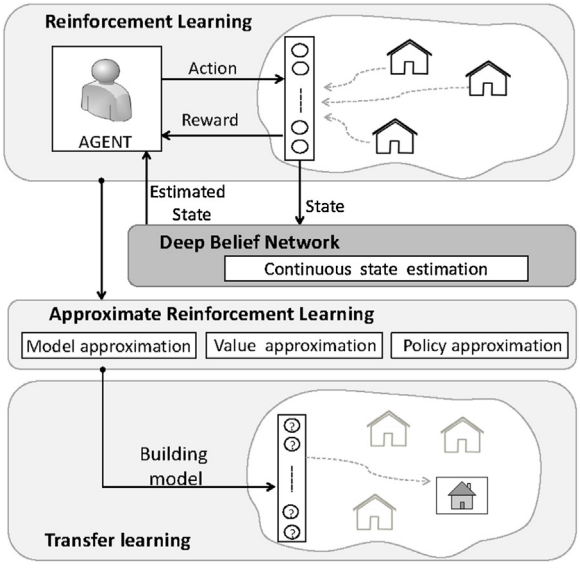

Fig. 1. The unsupervised learning explore and extends reinforcement and transfer learning setup, by including a Deep Belief Network for continuous states estimation.

given to the question: *How to estimate a continuous state space?* The idea is to find a lower-dimensional representation of the energy consumption data that preserves the pairwise distances as well as possible.

More formally, the energy prediction using unlabeled data problem presented in this paper is divided into three different sub-problems, namely:

1. Continuous state estimation problem: **Given** a dataset, $\mathcal{D} : \mathrm{IR} \rightarrow \mathcal{S}$, **find** a confined space state representation $\mathcal{S}_1$.
2. Reinforcement learning problem: **Given** a building model $M_1 = \langle \mathcal{S}_1, \mathcal{A}_1, \mathcal{T}.(\cdot,\cdot), \mathcal{R}_1 \rangle$, **find** a optimal policy, π_1^*.
3. Transfer learning problem: **Given** a model, $M_1 = \langle \mathcal{S}_1, \mathcal{A}_1, \mathcal{T}.(\cdot,\cdot), \mathcal{R}_1 \rangle$, a reasonable π_1^* and $M_2 = \langle \mathcal{S}_2, \mathcal{A}_2, \mathcal{T}.(\cdot,\cdot), \mathcal{R}_2 \rangle$, **find** a good π_2.

The proposed solution is presented in Section 4, where a new method to estimate continuous states in reinforcement learning using Deep Belief Network is detailed. Further this state estimation method is integrated in SARSA and Q-learning algorithms in order to improve the prediction accuracy.

3. Reinforcement learning

Reinforcement learning [28] is a field of machine learning inspired by psychology, which studies how artificial agents can perform actions in an environment to achieve a specific goal. Practically, the agent has to control a dynamic system by choosing actions in a sequential fashion. The dynamic system, known also as the environment, is characterized by states, its dynamics, and a function that describes the state's evolution given the actions chosen by the agent. After it executes an action, the agent moves to a new state, where it receives a reward (scalar value) which informs it how far it is from the goal (the final state). To achieve the goal, the agent has to learn a strategy to select actions, dubbed policy in the literature, in such a way that the expected sum of the rewards is maximized over time. Besides that, a state of the system captures all the information required to predict the evolution of the system in the next state, given an agent action. Also, it is assumed that the agent could perceive the state of the environment without error, and it could make its current decision based on this information. There are two different categories of RL algorithms, (i) Online RL wich are *interaction based* algorithms, such as Q-learning [26], SARSA [25] or Policy Gradient, and (ii) Offline RL, like Least-Square Policy Iteration or fitted Q-iteration. For a more comprehensive discussion of RL algorithms we refer to [29]. In the remaining of this paper will refer just to online RL.

A RL problem can be formalized using Markov decision process (MDPs). MDPs are defined by a 4-tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}.(\cdot,\cdot), \mathcal{R}.(\cdot,\cdot) \rangle$, where $\mathcal{S}$ is a set of states, $\forall s \in \mathcal{S}$, $\mathcal{A}$ is a set of actions, $\forall a \in \mathcal{A}$, $\mathcal{T}: \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is the transition function given by the probability that by choosing action a in state s at time t the system will arrive to state s' at time $t+1$, such that $p_a(s, s') = p(s_{t+1} = s' | s_t = s, a_t = a)$, and $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$ is the reward function, were $\mathcal{R}_a(s, s')$ is the immediate reward (or expected immediate reward) received by the agent after it performs the transition to state s' from state s. An important property in MDPs is the Markov property which makes the assumption that the state transitions are dependent just on the last state of the system, and are independent of any previous environment states or agent actions, i.e. $p(s_{t+1} = s', r_{t+1} = r | s_t, a_t)$ for all s', r, s_t, and a_t. The MDPs theory does not assume that $\mathcal{S}$ or $\mathcal{A}$ are finite, but the traditional algorithms make this assumption. In general, they can be solved by using linear or dynamic programming. The interested reader is referred to [30] for a more comprehensive discussion about MDPs. Furthermore, in the real-world, the states transitions probabilities $\mathcal{T}.(\cdot;\cdot)$ and the rewards $R.(\cdot;\cdot)$ are unknown, and the states space $\mathcal{S}$ or the actions space $\mathcal{A}$ might be continuous. Thus, RL represents a normal extension and generalization over MDPs for such situations, where the tasks are too large or too ill-defined, and can not be solved using optimal-control theory [25].

3.1. Q-learning

First, the Q-learning algorithm [26] is recommended like a standard solution in RL were the rules are often stochastic. This algorithm therefore has a function which calculates the Quality of a state-action combination, define by $Q : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$. Before learning has started, Q matrix returns an initial value. Then, each time the agent selects an action, and observes a reward and a new state that both may depend on the previous state and the selected action. The action-value function of a fixed policy π with the value function $V^\pi : \mathcal{S} \rightarrow \mathbb{R}$ is

$$Q^\pi(s, a) = r(s, a) + \gamma \sum_{s'} p(s'|s, a) V^\pi(s'), \quad \forall s \in \mathcal{S},\ a \in \mathcal{A} \tag{1}$$

The value of state-action pairs, $Q^\pi(s, a)$, represents the expected outcome when one agent is starting from s, executing a and then following the policy π afterwards, such that $V^\pi(x) = Q^\pi(x, \pi(x))$, with their corresponding Bellman equation

$$Q^*(s, a) = r(s, a) + \gamma \sum_{s'} p(s'|s, a) \max_b Q^*(s, b) \tag{2}$$

where the discount factor $\gamma \in [0, 1]$ trades off the importance of rewards and $b = \max(a)$. Thus, the optimal value are obtained for $\forall s \in S, V^*(s) = \max_a Q^*(s, a)$ and $\pi^*(s) = arg\max_a Q^*(s, a)$. The value of state-action pairs is given by the same formal expectation value, $\mathbb{E}_\pi$, of an expected total return r_t, such that $Q(s, a) = \mathbb{E}_\pi(r_t | s_t = s, a_t = a)$. The off-policy Q-learning algorithm have the update rule define by

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha_t [r_{t+1} + \gamma \max Q_t(s_{t+1}, a) - Q_t(s_t, a_t)] \tag{3}$$

where r_{t+1} is the reward observed after performing a_t in s_t, and where $\alpha_t(s, a)$, with all $\alpha \in [0, 1]$, is the learning rate which may be the same for all pairs. Q-learning algorithm has problems with big numbers of continuous states and discrete actions. Usually, it needs function approximations, e.g. neural networks, to associate triplets like (state, action, Q-value). Exploration of one MDP can be done under Markov assumption, to take into account just current state and action, but because in the real world we have Partially Observable MDPs, we may have better results if an arbitrary k number of history states and actions $(S_{t-k}, a_{t-k}, \cdots, S_{t-1}, a_{t-1})$ will be considerate [31], to clearly identify a triplet $\langle \mathcal{S}_t, \mathcal{A}_t, Q_t \rangle$ at time t.

3.2. SARSA

An interesting variation for Q-learning is the State-Action-Reward-State-Action (SARSA) algorithm [25], which aims at using Q-learning as part of a Policy Iteration mechanism. The major difference between SARSA and Q-Learning, is that the maximum reward for the next state in SARSA is not necessarily used for updating the Q-values. Therefore, the core of the SARSA algorithm is a simple value iteration update. The information required for the update is a tuple $(s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1})$, and the update is defined by

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha_t [r_{t+1} + \gamma Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t)] \tag{4}$$

where r_{t+1} is the reward and $\alpha_t(s, a)$ is the learning rate. In practice, Q-learning and SARSA are the same if we use a greedy policy (i.e. the agent chooses the best action always), but are different when the ϵ-greedy policy is used, which favors more random exploration.

In traditional reinforcement learning algorithms, only MDPs with finite states and actions are considered. However, building energy consumption can take nearly arbitrary real value resulting in a very large number of states in MDPs. Due to the fact that energy consumption can be seen as a time series problem, an prior discretization of the states space is not very useful. So, we try to find algorithms that work well with large (or continuous) states spaces, as it is shown next.

4. States estimation via Deep Belief Networks

Deep architectures [27] showed very good results in different applications, such as to perform non-linear dimensionality reduction [32], images recognition [33], video sequences, or motion-capture data [34]. A comprehensive analysis on dimensionality reduction and deep architectures can be referred to [35]. Overall, Deep Belief Networks (DBN) could be a way to naturally decompose the problem into sub-problems associated with different levels of abstraction.

4.1. Restricted Boltzmann Machines

DBNs are composed of several Restricted Boltzmann Machines (RBMs) stacked on top of each other [36]. A RBM is a stochastic recurrent neural network that consists of a layer of visible units, $\mathbf{v}$, and a layer of binary hidden units, $\mathbf{h}$. The total energy of the joint configuration of the visible and hidden units ($\mathbf{v}$, $\mathbf{h}$) is given by:

$$E(\mathbf{v}, \mathbf{h}) = -\sum_{i,j} v_i h_j W_{ij} - \sum_i v_i a_i - \sum_j h_j b_j \tag{5}$$

where i represents the indices of the visible layer, j those of the hidden layer, and $w_{i,j}$ denotes the weight connection between the ith visible and jth hidden unit. Further, v_i and h_j denote the state of the ith visible and jth hidden unit, respectively, a_i and b_j represent the biases of the visible and hidden layers. The first term, $\sum_{i,j} v_i h_j W_{ij}$ represents the energy between the hidden and visible units with their associated weights. The second, $\sum_i v_i a_i$ represent the energy in the visible layer, while the third term represents the energy in the hidden layer. The RBM defines a joint probability over the hidden and visible layer $p(\mathbf{v}, \mathbf{h})$

$$p(\mathbf{v}, \mathbf{h}) = \frac{e^{-E(\mathbf{v},\mathbf{h})}}{Z} \tag{6}$$

where Z is the partition function, obtained by summing the energy of all possible ($\mathbf{v}$, $\mathbf{h}$) configurations, $Z = \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}$. To determine the probability of a data point represented by a state v, the marginal probability is used, summing out the state of the hidden layer, such that $p(v) = \sum_h p(\mathbf{v}, \mathbf{h})$.

The above equation can be used for any given input to calculate the probability of either the visible or the hidden configuration to be activated. This values are further used to perform inference in order to determine the conditional probabilities in the model. To maximize the likelihood of the model, the gradient of the log-likelihood with respect to the weights must to be calculated. The gradient of the first term, after some algebraic manipulations can be written as

$$\frac{\partial \log(\sum_h \exp(-E(v, h)))}{\partial W_{ij}} = v_i \cdot p(h_j = 1|v) \tag{7}$$

However, computing the gradient of the second term is intractable.

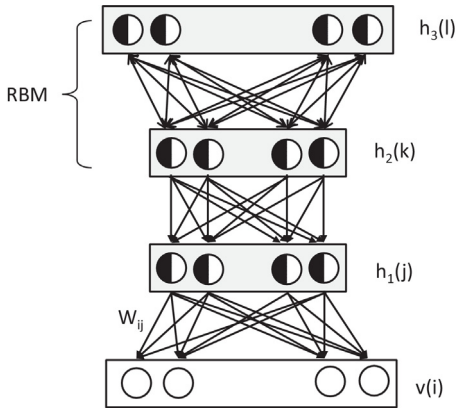

Fig. 2. A general Deep Belief Network structure with three hidden layers. The top two layers have undirected connections and form an associative memory, where ◐ denotes binary neurons and ○ represents the real values.

The inference of the hidden and visible layers in RBM can be done accordingly with the next formulas

$$p(\mathbf{h}_j = 1|\mathbf{v}) = \sigma(b_j + \sum_i v_i W_{ji}) \tag{8}$$

$$p(\mathbf{v}_i = 1|\mathbf{h}) = \sigma(a_i + \sum_j h_j W_{ji}) \tag{9}$$

where $\sigma(.)$ represents the sigmoid function. Moreover, to learn an RBM we can use the following learning rule which performs stochastic steepest ascent in the log probability of the training data [37]:

$$\frac{\partial \log(p(\mathbf{v}, \mathbf{h}))}{\partial W_{ij}} = \langle v_i h_j \rangle_0 - \langle v_i h_j \rangle_\infty \tag{10}$$

where $\langle \cdot \rangle_0$ denotes the expectations for the data distribution (p_0), and $\langle \cdot \rangle_\infty$ denotes the expectations under the model distribution.

4.2. Deep Belief Networks

Overall, a Deep Belief Network [27] is given by an arbitrary number of RBMs stack on the top of each other. This yields a combination between a partially directed and partially undirected graphical model. Therein, the joint distribution between visible layer x (input vector) and the l hidden layers h^k is define as follows

$$p(\mathbf{x}, \mathbf{h}^1, \ldots, \mathbf{h}^k) = \prod_{k=0}^{l-2} \mathrm{P}(\mathbf{h}^k|\mathbf{h}^{k+1}) \mathrm{P}(\mathbf{h}^{l-1}, \mathbf{h}^l) \tag{11}$$

where $\mathrm{P}(\mathbf{h}^k|\mathbf{h}^{k+1})$ is a conditional distribution for the visible units conditioned on the hidden units of the RBM at level k, and $\mathrm{P}(\mathbf{h}^{l-1}, \mathbf{h}^l)$ is the visible-hidden joint distribution in the top-level RBM. An example of a DBN with 3 hidden layers (i.e. $h_1(j)$, $h_2(k)$, and $h_3(l)$) is depicted in Fig. 2.

The top level RBM in a DBN acts as a complementary prior from the bottom level directed sigmoid likelihood function. A DBN can be trained in a greedy unsupervised way, by training separately each RBM from it, in a bottom to top fashion, and using the hidden layer as an input layer for the next RBM [38]. Furthermore, the DBN, can be used to project our initial states acquired from the environment to another state space with binary values, by fixing the initial states in the bottom layer of the model, and inferring the top hidden layer from them. In the end, the top hidden layer can be directly incorporated into the SARSA or Q-learning algorithms, as it is described in Algorithm 1.

Algorithm 1. *RL* extension including a DBN for state estimation.

```
%%DBN for state estimation
Initialize DBN
Initialize training set X with the states
for each RBM_k in DBN
  repeat training epoch
    For each training instance x ∈ X
      Set RBM_k^visible = x
      Run Markov chain in RBM_k
      Get Statistics for RBM_k
      Update Weights for RBM_k
    end for
  Until converge
  for each training instance x ∈ X
    Set RBM_k^visible = x
    Infer RBM_k^hidden
    Replace x in X with RBM_k^hidden
  end for
end for
%% Use the last computed X as states for RL(·)
%% RL(1): SARSA algorithm
Initialize Q(s, a) arbitrarily, where s ∈ X
repeat (for each episode)
  Initialize s
  Choose a from s using policy derived from Q
  repeat (for each step of episode)
    Take action a, observe r, s'
    Choose a' from s' using policy derived from Q
    Q(s, a) ← Q(s, a) + α[r + γQ(s', a') − Q(s, a)]
    s ← s'
    a ← a'
  until s is terminal
until Q optimal
%% RL(2): Q-learning algorithm
Initialize Q(s, a) arbitrarily, where s ∈ X
repeat (for each episode)
  Initialize s
  repeat (for each step of episode)
    Choose a from s using policy derived from Q
    Take action a, observe r, s'
    Q(s, a) ← Q(s, a) + α[r + γ max_α Q(s', a') − Q(s, a)]
    s ← s'
  until s is terminal
  untilQ optimal
```

Now that we have considered the problem of state estimation and we incorporated all three sub-problems in a unified approach we look into the experimental validation.

5. Data set characteristics

The proposed solution is experimentally evaluated using a dataset recorded over seven years, more exactly between January 6, 2007 and January 31, 2014. The loads profiles, including different residential and commercial buildings, are made available by Baltimore Gas and Electric Company [39]. For every type of building analyzed the available historical load data in kWh represent an average building profile per hour. Overall, there are five different buildings profiles, as presented in Table 1.

For a more comprehensive view of the datasets used in this paper we have shown in Fig. 3 the hourly evolution of the electrical energy consumption for a General Service (G) dataset, including

Table 1
Building types in datasets.

Residential	R	Residential (non-electric heat)
	R (ToU)	Residential Time-of-Use (non-electric heat)
	RH	Residential (electric heat)
	RH (ToU)	Residential Time-of-Use (electric heat)
Commercial	G	General Service (< 60 kW), Commercial, Industrial & Lighting

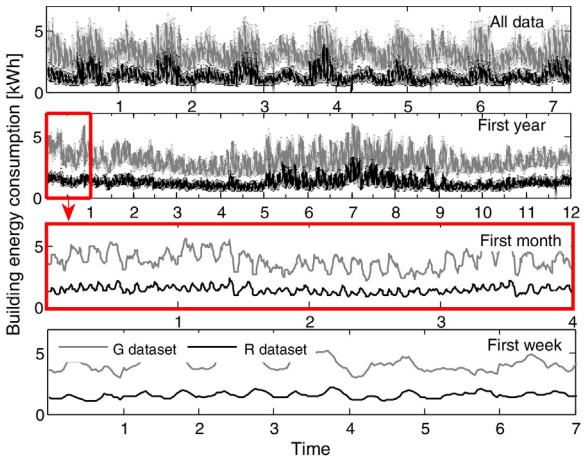

Fig. 3. Electrical energy consumption for a Commercial, Industrial & Lighting (G) dataset and for a residential building with non-electric heat (R) building over different time horizons.

Commercial, Industrial & Lighting, and a residential with non-electric heat (R) building over different time horizons. Moreover, some general characteristics for the entire dataset are graphically depicting in Fig. 4. In all experiments the data were separated into the training and testing datasets. More precisely, the data collected from 1st June 2007 until 1st January 2013 (2041 days) were used in the learning phase and the remaining data, between January 2013 and 31 January 2014 (396 days) were used to evaluate the performance of the methods. The metrics used to assess the quality of the different buildings energy consumption prediction are described further.

5.1. Metrics for prediction assessment

As we mention earlier, the goal is to achieve good generalization by making accurate prediction for new building energy consumption data. Firstly, some quantitative insight into the dependence of the generalization performance of our approach are evaluated using the root-mean-square error defined by $RMSE = \sqrt{(1/N)\sum_{i=1}^{N}(v_i - \hat{v}_i)^2}$, where N represents the number of multi-steps prediction within a specified time horizon, v_i represents the real values for the time-step i and $\hat{v}_i$ represents the model estimated value at the same time-step. Then, by using the Pearson product-moment correlation coefficient (R), insights are given on the degree of linear dependence between the real value and the predicted value. Hence $R(u, v) = (\mathbb{E}[(u - \mu_u)(v - \mu_v)]/\sigma_u\sigma_v)$, where $\mathbb{E}[\cdot]$ is

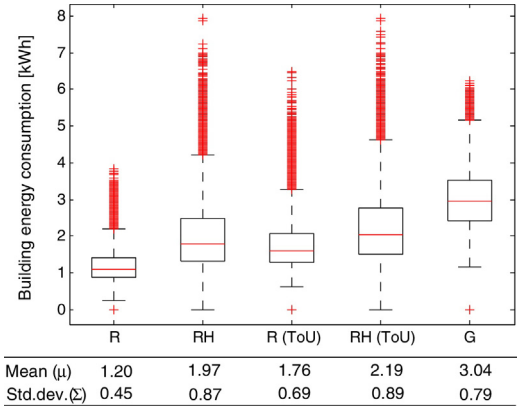

Fig. 4. General characteristics of all data sets: a box-plot with the exactly value for mean and standard deviation encoded in it.

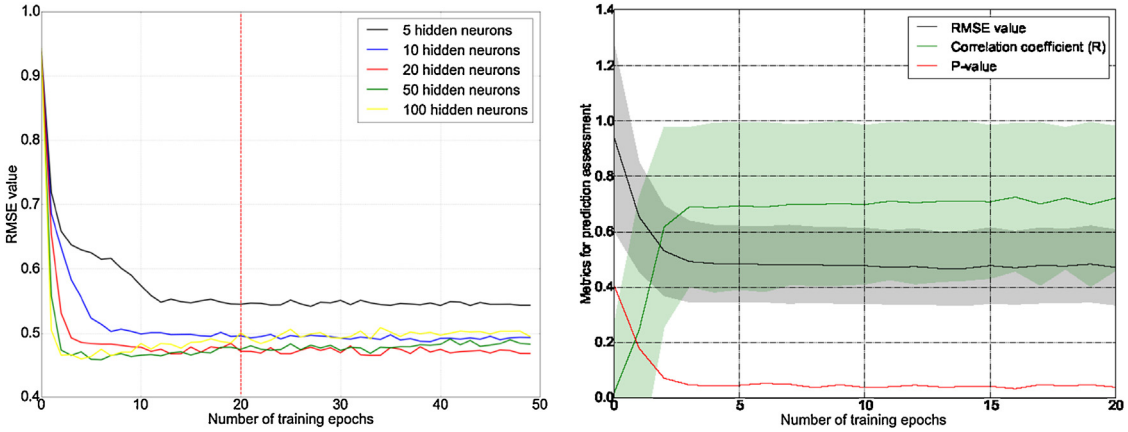

Fig. 5. (Left) The RMSE values observed for different RBM configurations in the DBN architecture, with varying number of hidden neurons, as a function of training epochs. (Right) Performance metrics for the chosen RBM configuration with 10 hidden neurons.

the expected value operator with standard deviations σ_u and σ_v. The correlation coefficient may take on any value within the range $[-1,1]$. The sign of the correlation coefficient defines the direction of the relationship, either positive or negative. Finally, we perform the Kolmogorov–Smirnov test [40] in order to gain insights on statistical significance of our results. The Kolmogorov–Smirnov test has the advantage of making no assumption about the distribution of data. This elaborate statistical test is not a typically metric used in the analyze of the prediction accuracy, but is imposed by the fact that the learning and the testing procedure is made using different building types. Hence, exceeding the statistical significance level, $p < 0.05$, would be expected and will validate the different probability distribution function from where this data are provided.

6. Empirical results

To assess the performance of our extended reinforcement and transfer learning approaches presented in Section 4, we have designed five different scenarios. These are selected to cover various multi-step prediction at different resolution, and are summarized in Table 2. Further on, before to go in the deep analyses of the numerical results, firstly we present some details of the implementation.

6.1. Implementations details

The implementation has been done in two parts. Firstly, a DBN is implemented, and secondly the RL algorithms use the DBN in their implementations for continuous states estimation, as it is shown next.

6.1.1. Continuous states estimations using DBN

We implemented the DBN in MATLAB® from scratch using the mathematical details described in Section 4. In order to obtain a good prediction we investigate carefully the choice of the optimal number of hidden units in our DBN configuration with respect to the RMSE evolution, see Fig. 5.

Table 2
Summary of the experiments.

	Notation	Time horizon	Resolution
Scenario 1	S1	1 h	1 h average
Scenario 2	S2	1 day	1 h average
Scenario 3	S3	1 week	1 h average
Scenario 4	S4	1 month	1 h average
Scenario 5	S5	1 year	1 week average

Thus, the number of hidden neurons was set to 10 and the learning rate was 10^{-3}. The momentum was set to 0.5 and the weight decay to 0.0002. We trained the model for 20 epochs, but as it can be seen in Fig. 5 (right), the model converged after approximately 4 epochs. More details about the optimal choice of the parameters can be found in [41].

6.1.2. SARSA and Q-learning

We implemented the SARSA and Q-learning in MATLAB® using the mathematical details described in Section 3. In both cases the learning rate was set to 0.4 and the discount factor was considerate 0.99. Both parameters have a direct influence on the performance of the both algorithms.

The choice of these parameters was made after a thorough examination of the RMSE outcome, as is shown for example in Fig. 6. Overall, the learning rate determines to what extent the newly acquired information will override the old information and the discount factor determines the importance of future rewards, for example $\gamma = 0$ will make the agent "opportunistic" by only considering current rewards, while a discount factor approaching 1 will make it strive for a long-term high reward.

6.2. Numerical results

In this section, we test and illustrate the two unsupervised learning approaches using the data set described in Section 5. Different scenarios, as summarized in Table 2, have been created to assess the performance of the proposed models.

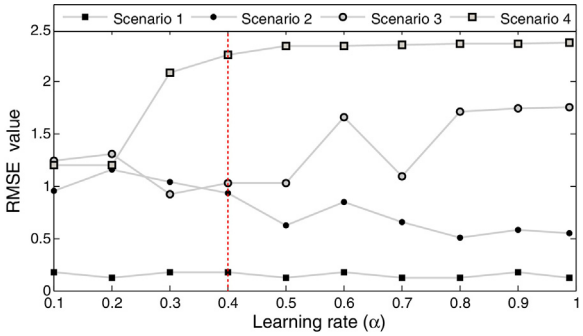

Fig. 6. Analyses of RMSE values obtained from different α value in exploration step, for different scenarios. This involve the prediction of G dataset (Commercial, Industrial & Lighting consumption, General Service (< 60 kW)).

Table 3
Using Commercial, General Service (G) (< 60 kW) dataset to predict rezidential energy consumption, such as R, R (ToU), RH and RH (ToU) values using SARSA, Q-learning, SARSA with DBN extension and Q-learning with DBN extension.

	Method	G			R			R (ToU)			RH			RH (ToU)		
		RMSE	*R*	*p*-Val	RMSE	*R*	*p*-Val	RMSE	*R*	*p*-Val	RMSE	*R*	*p*-Val	RMSE	*R*	*p*-Val
S1	SARSA	0.18	0.90	5.6e−10	0.02	0.99	2.8e−23	0.10	0.93	3.9e−12	0.36	0.91	1.2e−10	0.42	0.88	4.2e−09
	Q-learning	0.22	0.86	2.2e−08	0.02	0.99	2.8e−23	0.04	0.99	2.5e−21	0.34	0.92	3.3e−11	0.34	0.92	3.3e−11
	SARSA + DBN	0.04	0.01	1.7e−05	0.02	0.99	3.4e−20	0.06	0.98	4.7e−14	0.04	0.99	1.2e−23	0.04	0.01	7.8e−26
	Q-learning + DBN	0.01	0.99	6.9e−38	0.03	0.97	7.1e−16	0.09	0.94	2.1e−12	0.04	0.99	1.1e−23	0.02	0.99	5.3e−33
S2	SARSA	0.65	0.36	0.0097	0.75	0.52	1.3e−04	0.47	0.42	0.0029	1.23	0.55	4.3e−05	1.20	0.65	3.5e−07
	Q-learning	1.09	0.17	0.2460	0.98	0.30	0.0358	0.40	0.81	1.4e−12	1.28	0.52	1.4e−04	1.55	0.41	0.0038
	SARSA + DBN	0.38	0.65	1.3e−05	0.37	0.98	0.39	0.37	0.79	0.0014	0.46	0.81	0.0023	0.47	0.67	2.6e−05
	Q-learning + DBN	0.33	0.84	6.2e−14	0.37	0.08	0.5539	0.29	0.74	1.7e−09	0.41	0.50	2.3e−04	0.66	0.26	0.0671
S3	SARSA	1.27	0.12	0.0877	1.73	0.21	0.0026	1.36	0.27	1.1e−04	1.59	0.29	2.6e−05	1.33	0.26	2.3e−04
	Q-learning	1.39	0.09	0.2052	1.10	0.24	7.6e−04	0.83	0.23	0.0012	1.47	0.25	3.1e−04	1.61	0.06	0.3589
	SARSA + DBN	0.69	0.90	1.4e−05	1.31	0.99	0.88	0.55	0.98	0.1623	1.33	0.99	0.7896	1.18	0.99	0.5244
	Q-learning + DBN	0.62	0.38	4.8e−08	0.98	0.11	0.0978	0.58	0.30	2.1e−05	1.26	0.12	0.0950	1.30	0.03	0.5932
S4	SARSA	1.55	0.09	0.0128	3.70	−0.25	1.3e−11	2.39	0.07	0.0361	2.05	0.07	0.0397	1.89	0.16	1.0e−05
	Q-learning	1.41	0.15	6.1e−05	1.24	0.07	0.0404	1.14	−0.04	0.2000	1.67	0.08	0.0309	1.71	0.02	0.4621
	SARSA + DBN	1.14	0.98	2.2e−04	1.45	0.99	0.29	1.17	0.98	0.0025	1.33	0.99	−8.51e−5	1.21	0.99	0.1347
	Q-learning + DBN	0.98	0.34	2.5e−20	1.40	0.01	0.8960	0.87	−0.13	5.2e−04	1.52	0.11	0.0022	1.55	0.17	3.2e−06
S5	SARSA	1.01	−0.08	0.5419	2.61	−0.15	0.2484	2.04	−0.20	0.1298	2.16	−0.31	0.0197	1.95	−0.29	0.0276
	Q-learning	0.72	0.30	0.0208	2.28	−0.20	0.1334	1.81	−0.20	0.1267	1.83	−0.33	0.0125	1.59	−0.08	0.5542
	SARSA + DBN	0.05	0.65	1.4e−08	0.08	0.66	5.3e−09	0.10	0.74	6.4e−12	0.11	0.89	6.02e−22	0.24	0.48	8.7e−05
	Q-learning + DBN	0.03	0.02	0.8245	0.02	0.37	0.0031	0.02	0.37	0.0028	0.03	0.22	0.0873	0.03	0.06	0.6315

* Statistical significant at $p < 0.05$.

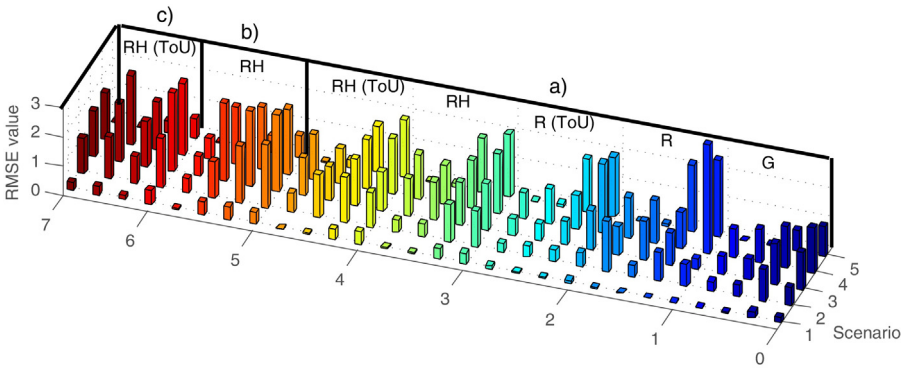

Fig. 7. Overview of errors obtained, where (a) using G dataset we predict R, R(ToU), RH and RH(ToU) values. (b) Using R we predict RH and (c) using R(ToU) we predict the RH(ToU). Four methods are used: SARSA, Q-learning, SARSA with DBN extension and Q-learning with DBN extension.

6.2.1. Commercial to residential transfer

In this set of experiments, we use Commercial, Industrial & Lighting data to train the DBN model. Furthermore, we use the trained DBN model to predict four different types of unseen residential building consumption, such as residential with electric heat and without electric heat, and residential electric consumption with TOU pricing, as it is shown in Table 3 and Fig. 7. The analysis of the different types of residential buildings advances the insight on the generalization capabilities of our proposed method and studies its robustness by testing the behaviour on different probability distributions (see Fig. 4).

6.2.2. Residential to residential transfer

During these experiments we learn and transfer one type of residential building energy demand profile to another type of residential building with different characteristics. More exactly, we used to train the learning algorithm (i) A residential building profile without electric heat (R), and (ii) A residential building with electric heat (RH). The prediction results of these two building models can be seen in Tables 4 and 5.

In Tables 3–5, the RMSE values show a good agreement between the real values and the model estimated values. In addition, the confidence in our results is formally determined not just by the RMSE values, but also by the correlation coefficient and the number of steps predicted into the future. For example, if there is just one step ahead, such as in Scenario 1, then the Pearson correlation coefficient needs to be very close to 1 or −1 in order to consider it statistically significant. However, in the case of Scenarios 3 and 4, where the prediction is made on 168 and 672 future steps, a coefficient close to 0 can still be considered highly significant. More discussions about the robustness of the correlation coefficient can be found in [42]. Still, the inaccuracy was reflected in a negative correlation coefficient in 24% of the experiments when we used the simple form of the SARSA and Q-learning methods. By contrast, our two improved approaches, SARSA with DBN extension and Q-learning with DBN extension, shows a negative correlation in just 4% of the cases. Overall, the Kolmogorov–Smirnov test in most cases confirms that the data do indeed come from different distributions. This is partially due to the unique characteristics of this dataset, given by the presence of a highly non-linear profile shape and large outlier values, as seen in Fig. 4. All of these observations, give a strong argument for employing a more comprehensive examination of the distributions used in the transfer learning. Nevertheless, the results presented in Tables 3–5, demonstrate that the energy prediction accuracies in terms of RMSE significantly improve in 91.42% of the cases after using a DBN for automatically

Table 4
Prediction of residential building with electric heat consumption using data collected from a residential with non-electric heat building.

	Methods	RMSE	*R*	*p*-Value
Scenario 1	SARSA	0.42	0.88	4.2e−09
	Q-learning	0.44	0.87	1.1e−08
	SARSA with DBN	0.42	0.88	5.8e−09
	Q-learning with DBN	0.03	0.99	7.2e−27
Scenario 2	SARSA	2.15	−0.18	0.2175
	Q-learning	1.93	−0.10	0.4802
	SARSA with DBN	1.25	0.61	3.7e−06
	Q-learning with DBN	0.5	0.64	9.2e−07
Scenario 3	SARSA	2.63	−0.27	8.6e−05
	Q-learning	2.57	−0.18	0.0094
	SARSA with DBN	2.67	0.13	0.06
	Q-learning with DBN	0.69	0.09	0.1863
Scenario 4	SARSA	2.23	0.04	0.2504
	Q-learning	2.14	0.11	0.0015
	SARSA with DBN	1.97	−0.09	0.01
	Q-learning with DBN	0.71	−0.10	0.0072
Scenario 5	SARSA	0.74	0.62	2.8e−07
	Q-learning	0.57	0.62	2.1e−07
	SARSA with DBN	0.03	0.43	4.8e−04
	Q-learning with DBN	0.02	0.51	0.0259

Table 5
Prediction of residential building consumption with electric heat using data collected from a residential with non-electric heat building, boht with ToU pricing.

	Methods	RMSE	*R*	*p*-Value
Scenario 1	SARSA	0.50	0.83	1.8e−07
	Q-learning	0.16	0.99	1.7e−25
	SARSA with DBN	0.28	0.94	6.1e−13
	Q-learning with DBN	0.24	0.99	2.0e−21
Scenario 2	SARSA	1.69	0.33	0.0200
	Q-learning	0.91	0.83	3.0e−13
	SARSA with DBN	1.42	0.55	4.09e−05
	Q-learning with DBN	1.18	0.77	1.0e−10
Scenario 3	SARSA	2.69	−0.11	0.1205
	Q-learning	1.65	0.17	0.0031
	SARSA with DBN	1.98	0.27	1.2e−04
	Q-learning with DBN	1.55	0.21	0.0167
Scenario 4	SARSA	2.45	−0.01	0.9477
	Q-learning	1.62	0.17	3.7e−06
	SARSA with DBN	2.38	0.24	4.7e−11
	Q-learning with DBN	1.60	0.28	3.3e−14
Scenario 5	SARSA	0.67	0.19	0.0014
	Q-learning	0.41	0.47	2.0e−04
	SARSA with DBN	0.03	0.34	0.006
	Q-learning with DBN	0.02	0.42	6.4e−04

computing high-level features from the unlabelled data, as compared to the situation when the counterpart RL methods are used without any DBN extension.

Notably, the proposed approach is also suitable when we have access to historical data. In the scope of this argument, the result obtained in first column of Table 3 are expected to be equivalent with the results obtain with any supervised learning methods, such as ANN or SVM like. Nevertheless, the RMSE accuracy obtained using the Q-learning algorithm with the DBN extension for the long-term forecasting of buildings energy consumption (Scenario 5) is greater than 90% in all the experiments than Q-learning without DBN extension. For example, in Table 4 the RMSE is 0.02 if we use Q-learning with DBN versus 0.57 for Q-learning without DBN, yielding to a 96.5% improved RMSE accuracy.

7. Discussion and conclusion

In this paper, a new paradigm on building energy prediction has been introduced, which does not require historical data from the specific building under scrutiny. In a unified approach, we can successfully learn a building model by including a generalization of the state space domain, then we transfer it across other building. The contribution is twofold. First, we present a Deep Belief Network for automatically feature extraction and second, we extend two standard reinforcement learning algorithms able to perform knowledge transfer between domains (buildings models), namely State-Action-Reward-State-Action (SARSA) algorithm and Q-learning algorithm by incorporating the states estimated with the Deep Belief Network. The novel proposed machine learning methods for energy prediction are evaluated over different time horizons with different time resolutions using real data. Notably, it can be observed that as the prediction horizon is increasing, SARSA and Q-learning extensions by including a DBN for states estimation seem to be more robust and their prediction error is approximately 20 times lower that of their unextended versions. The strength of this method is given by the DBN generalization capabilities over the underlying state space for a new building and the robustnesses to invariance in the states representation. However, a forthcoming deep investigation can be done at different Smart Grid levels in order to help the transition to the future energy system.

Acknowledgements

This research has been funded by NL Enterprise Agency RVO.nl under the TKI Switch2SmartGrids project of Dutch Top Sector Energy.

References

[1] L. Yang, H. Yan, J.C. Lam, Thermal comfort and building energy consumption implications – a review, Appl. Energy 115 (2014) 164–173, http://dx.doi.org/10.1016/j.apenergy.2013.10.062.
[2] E.A. Bakirtzis, C.K. Simoglou, P.N. Biskas, D.P. Labridis, A.G. Bakirtzis, Comparison of advanced power system operations models for large-scale renewable integration, Electr. Power Syst. Res. 128 (2015) 90–99, http://dx.doi.org/10.1016/jepsr.2015.06.025.
[3] A. Costa, M.M. Keane, J.I. Torrens, E. Corry, Building operation and energy performance: monitoring, analysis and optimisation toolkit, Appl. Energy 101 (2013) 310–316, http://dx.doi.org/10.1016/j.apenergy.2011.10.037.
[4] M. Simoes, R. Roche, E. Kyriakides, S. Suryanarayanan, B. Blunier, K. McBee, P. Nguyen, P. Ribeiro, A. Miraoui, A comparison of smart grid technologies and progresses in Europe and the U.S., IEEE Trans. Ind. Appl. 48 (4) (2012) 1154–1162, http://dx.doi.org/10.1109/TIA.2012.2199730.
[5] X. Li, D. Gong, L. Li, C. Sun, Next day load forecasting using SVM, in: J. Wang, X.-F. Liao, Z. Yi (Eds.), Advances in Neural Networks – ISNN 2005, Vol. 3498 of Lecture Notes in Computer Science, 2005.
[6] W.-C. Hong, Electric load forecasting by support vector model, Appl. Math. Model. 33 (5) (2009) 2444–2454.
[7] S. Wong, K.K. Wan, T.N. Lam, Artificial neural networks for energy analysis of office buildings with daylighting, Appl. Energy 87 (2) (2010) 551–557.
[8] S.A. Kalogirou, Artificial neural networks in energy applications in buildings, Int. J. Low-Carbon Technol. 1 (3) (2006) 201–216.
[9] T. Mestekemper, G. Kauermann, M.S. Smith, A comparison of periodic autoregressive and dynamic factor models in intraday energy demand forecasting, Int. J. Forecast. 29 (1) (2013) 1–12.
[10] M. Wytock, J.Z. Kolter, Large-scale probabilistic forecasting in energy systems using sparse gaussian conditional random fields, in: Proceedings of the 52nd IEEE Conference on Decision and Control, CDC 2013, December 10–13, 2013, Firenze, Italy, 2013, pp. 1019–1024.
[11] E. Mocanu, P.H. Nguyen, M. Gibescu, W. Kling, Comparison of machine learning methods for estimating energy consumption in buildings, in: Proceedings of the 13th International Conference on Probabilistic Methods Applied to Power Systems (PMAPS), July 10–13, Durham, UK, 2014.
[12] M. Aydinalp-Koksal, V.I. Ugursal, Comparison of neural network, conditional demand analysis, and engineering approaches for modeling end-use energy consumption in the residential sector, Appl. Energy 85 (4) (2008) 271–296.
[13] L. Xuemei, D. Lixing, L. Jinhu, X. Gang, L. Jibin, A novel hybrid approach of KPCA and SVM for building cooling load prediction, in: Knowledge Discovery and Data Mining, 2010. Third International Conference on WKDD '10, 2010.
[14] L. Suganthi, A.A. Samuel, Energy models for demand forecasting – a review, Renew. Sustain. Energy Rev. 16 (2) (2012) 1223–1240.
[15] M. Krarti, Energy Audit of Building Systems: An Engineering Approach, Mechanical and Aerospace Engineering Series, 2nd ed., Taylor & Francis, 2012.
[16] A.I. Dounis, Artificial intelligence for energy conservation in buildings, Adv. Build. Energy Res. 4 (1) (2010) 267–299.
[17] A. Foucquier, S. Robert, F. Suard, L. Stéphan, A. Jay, State of the art in building modelling and energy performances prediction: a review, Renew. Sustain. Energy Rev. 23 (0) (2013) 272–288.
[18] H. xiang Zhao, F. Magoulès, A review on the prediction of building energy consumption, Renew. Sustain. Energy Rev. 16 (6) (2012) 3586–3592.
[19] D. Ernst, M. Glavic, F. Capitanescu, L. Wehenkel, Reinforcement learning versus model predictive control: a comparison on a power system problem, IEEE Trans. Syst. Man Cybern. B: Cybern. 39 (2) (2009) 517–529.
[20] R. Crites, A. Barto, Improving elevator performance using reinforcement learning, in: in: Advances in Neural Information Processing Systems, vol. 8, MIT Press, 1996, pp. 1017–1023.
[21] H. Ammar, D. Mocanu, M. Taylor, K. Driessens, K. Tuyls, G. Weiss, Automatically mapped transfer between reinforcement learning tasks via three-way restricted Boltzmann machines, in: in: Machine Learning and Knowledge Discovery in Databases, Vol. 8189 of Lecture Notes in Computer Science, 2013, pp. 449–464.
[22] B. Sallans, G.E. Hinton, Reinforcement learning with factored states and actions, J. Mach. Learn. Res. 5 (2004) 1063–1088.
[23] N. Heess, D. Silver, Y.W. Teh, Actor-critic reinforcement learning with energy-based policies, in: JMLR Workshop and Conference Proceedings: EWRL 2012, 2012.
[24] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, D. Hassabis, Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533.
[25] R.S. Sutton, A.G. Barto, Introduction to Reinforcement Learning, 1st ed., MIT Press, Cambridge, MA, USA, 1998.
[26] C.J.C.H. Watkins, P. Dayan, Technical note: Q-learning, J. Mach. Learn. Res. 8 (3-4) (1992) 279–292.
[27] Y. Bengio, Learning deep architectures for AI, Found. Trends Mach. Learn. 2 (1) (2009) 1–127, also published as a book. Now Publishers, 2009.
[28] M. Wiering, M. van Otterlo, Reinforcement Learning: State-of-the-Art, Springer, 2012.
[29] L. Busoniu, D. Ernst, B. De Schutter, R. Babuska, Approximate reinforcement learning: an overview, in: 2011 IEEE Symposium on Adaptive Dynamic Programming And Reinforcement Learning (ADPRL), 2011, pp. 1–8.
[30] M.L. Puterman, Markov Decision Processes: Discrete Stochastic Dynamic Programming, 1st ed., John Wiley & Sons, Inc., New York, NY, USA, 1994.
[31] M. Castronovo, F. Maes, R. Fonteneau, D. Ernst, Learning exploration/exploitation strategies for single trajectory reinforcement learning, in: in: EWRL, Vol. 24 of JMLR Proceedings, 2012, pp. 1–10.
[32] G. Hinton, R. Salakhutdinov, Reducing the dimensionality of data with neural networks, Science 313 (5786) (2006) 504–507.
[33] G.E. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets, Neural Comput. 18 (7) (2006) 1527–1554.
[34] G.W. Taylor, G.E. Hinton, S.T. Roweis, Two distributed-state models for generating high-dimensional time series, J. Mach. Learn. Res. 12 (2011) 1025–1068.
[35] L.J. van der Maaten, E.O. Postma, H.J. van den Herik, Dimensionality reduction: a comparative review, J. Mach. Learn. Res. 10 (1-41) (2009) 66–71.
[36] G.E. Hinton, S. Osindero, A fast learning algorithm for deep belief nets, Neural Comput. 18 (2006) 2006.
[37] G.E. Hinton, Training products of experts by minimizing contrastive divergence, Neural Comput. 14 (8) (2002) 1771–1800.

[38] R. Salakhutdinov, Learning deep boltzmann machines using adaptive MCMC, in: Proceedings of the 27th International Conference on Machine Learning (ICML-10), Haifa, Israel, 2010, pp. 943–950.
[39] B.G.E. Company. http://www.supplier.bge.com (last visit 17.10.15).
[40] F.J. Massey, The Kolmogorov–Smirnov test for goodness of fit, J. Am. Stat. Assoc. 46 (253) (1951) 68–78.
[41] G.E. Hinton, A practical guide to training restricted Boltzmann machines, in: in: Neural Networks: Tricks of the Trade, Lecture Notes in Computer Science, 2nd ed., Springer, 2012, pp. 599–619.
[42] J.R.K. Susan, J. Devlin, R. Gnanadesikan, Robust estimation and outlier detection with correlation coefficients, Biometrika 62 (3) (1975) 531–545.