Fitted Q-iteration and Functional Networks for Ubiquitous Recommender Systems

Matteo Gaeta[(1)], Francesco Orciuoli[(2)], Luigi Rarità[(3)], Stefania Tomasiello[(1,3)],
[(1)]Dipartimento di Ingegneria dell'Informazione,
Ingegneria Elettrica e Matematica Applicata,
University of Salerno, Via Giovanni Paolo II, 132,
84084, Fisciano (SA), Italy.
[(2)]Dipartimento di Scienze Aziendali - Management & Innovation Systems,
University of Salerno, Via Giovanni Paolo II, 132,
84084, Fisciano (SA), Italy.
[(3)]CO.RI.SA., COnsorzio RIcerca Sistemi ad Agenti, University of Salerno,
Via Giovanni Paolo II, 132,
84084, Fisciano (SA), Italy.

June 19, 2016

Abstract

Ubiquitous recommender systems facilitate users on–location by personalized recommendations of items in the proximity via mobile devices. Due to a high variability of situations and preferences, an efficient resource processing is needed in order to assist the user in a proper way. In this paper, we consider a recommender system, able to learn preferences/habits of users through contextual information, such as location and time, using a new offline model–free approximate Q–iteration. Following the basic idea of Fitted Q–iteration, the paper focuses on a computational scheme, based on Functional Networks and that, unlike the well–known neural ones, does not require a large number of training samples. A preliminary case study, which deals with a shopping mall, is useful to show the approximation capabilities of the proposed approach.

Keywords: Q–learning, Functional Networks, ubiquitous context, recommender systems

1 Introduction

Recommender Systems (RSs) are able to provide people with useful information for suitable purposes [1]. A limit of traditional RSs is that the available information is not all used to generate recommendations. Context–Aware Recommender Systems use contextual parameters in the recommendation processes to provide better results to users [2]–[4].

In these cases, knowledge about the environment has to be learned to avoid unwhished recommendations which could disturb the user in certain critical situations.

An important issue is how to obtain contextual information, related to the user, the computing system, the environment and the various interactions between the user and the system [5]. Due to the progress of mobile phone technologies, ubiquitous RS represent a possible answer to provide users with more personalized and context–aware recommendations on location, with emphasis on various types of contexts.

In [6] a new class of ubiquitous RSs, i.e. Ubiquitous Context–Aware Recommender Systems, is introduced. It represents a subset of Ubiquitous Recommender Systems via fusion of the usual contextual information with the one available via mobile devices, such as location.

In mobile RSs for indoor shopping, a simple rule–based recommendation algorithm is useful, e.g. to recommend certain brand stores to users, as a limited number of stores in a shopping mall are considered [7].

In outdoor environments, where more stores have to be considered, an agent based approach is used [8], even if the users' preferences are ignored and only products with lowest prices are recommended.

In these papers, the key elements for an unambiguously communication with the user are not exploited.

In conversational RSs, modeling the user interaction as a Markov Decision Process and employing Reinforcement Learning (RL) to learn the optimal policy (to respond the user actions) seems to be usual, also to facilitate an unambiguous communication[9]–[11].

In RL the learner interacts with its environment and receives reinforcements for its actions [12]. The optimal actions are the ones which maximize the expected cumulative reinforcement that the learner receives in the long run (by the end of the interaction).

A popular RL algorithm is Q–learning, which uses an action–value model, creating a function to deal with different states; it is conceived to determine the optimal policy in a step–by–step manner [13].

Recently, Q–learning has been used for web RSs in order to address the problem of information overload on the Internet, by directing users toward the resources that best meet their needs and interests [14].

For finite and small enough state and action spaces, the Q–function is represented in tabular form, so its approximation (in batch and in on–line mode) and the derived control policy are straightforward. This approach is not used for cases of continuous or large discrete state and/or action spaces. Besides, an approximation of the Q–function all over the state–action space has to be derived from finite and generally very sparse sets of four–tuples (state, action, immediate reward, successor state).

In order to respond to these issues, the Fitted Q–iteration (FQI) algorithm is proposed [15]. FQI is a batch mode RL algorithm which yields an approximation of the Q–function corresponding to an infinite horizon optimal control problem with discounted rewards, by iteratively extending the optimization horizon. At the first iteration the FQI algorithm uses the training set, with inputs as the state-action pairs and outputs as the observed rewards, with the aim of producing an approximation of the expected reward. In the subsequent iterations, only the output values are updated using the value of Q–function produced at the preceding step and information about the reward and the successor state reached in each tuple. Since all updates are done offline, approximating the Q–function is a separate, supervised learning problem. The question arises if there are function approximators especially suited for offline updating.

In [15] tree based regression methods are examined, whereas in [16] the use of multilayer–perceptrons is proposed. All these methods are model–free, i.e. only the transition tuples (state, action, immediate reward, successor state) need to be known.

In this paper another model–free approach is proposed, which has the main advantage of requiring few training samples. Our method combines FQI with Functional Networks (FNs), introduced by Castillo [17] to provide a new powerful alternative to Artificial Neural Networks (ANNs). The main feature of a FN is in the activation functions which are unknown multivariate ones from given families to be estimated during the learning process (in ANNs the neural functions are univariate and the weights have to be learned). Hence, arbitrary functional models are defined. It has been shown that every problem, which can be solved by an ANN, can also be formulated by a FN; the problems, which cannot be solved by an ANN, can be formulated by a FN [18].

The approach herein discussed has been successfully used to solve some continuous control problems [19].

Here we apply the proposed method for the first time to a problem in a discrete domain, i.e. an ubiquitous RS for products and services (such as restaurants), that is able, in addition to respond unambiguously to user's initiatives, to recommend items in the user's proximity and services at a certain time, while the user is in a shopping mall.

The method is suitable to disambiguate user's requests, since this enlarges the space of actions. Besides, it allows to learn the user's preference. In fact, based on the results, the context information is enriched by collecting data such as the items the user brings in each visit, the day and time of each purchase, in which kind of restaurant, day and time he/she eats, and so on.

The good performance of our method with a relatively small number of training samples is shown.

The paper is organized as follows. Section 2 provides a literature review on FQI and FNs. Section 3 introduces the proposed method. Section 4 is devoted to numerical examples. The paper ends with conclusions in Section 5.

2 Literature review

This section presents a literature review about the Fitted Q–iteration and Functional Networks.

2.1 Fitted Q–iteration

The Fitted Q–Iteration (FQI) derives from the pioneering work by Ormoneit and Sen [22], who combine the idea of fitted value iteration [23] with kernel based reinforcement learning. Hence, the Q–function determination problem is reformulated in terms of a sequence of kernel–based regression problems.

FQI is introduced by Ernst et al. [15] in order to fit an approximation procedure (of parametric or non–parametric type) for the Q–function, using a set of four–tuples. In their numerical experiments, the authors refer to tree–based algorithms. The identity function to the output is a special case that leads to FQI combined with basis functions, focusing on piecewise linear approximations.

For each step the algorithm uses either the function computed at the previous step or the full set of four–tuples obtained from the observation of the system. This allows determining a new training set which is useful to obtain the next function of the sequence via a supervised learning (regression) method. Hence, a sequence of Q_N–functions, which converges under suitable assumptions, is obtained.

Antonos et al. [24] propose a variant of FQI, where the greedy action selection is replaced by the search for a policy (within a restricted set of possible policies) through a maximization of the average action values.

For the case of continuous action spaces, Neumann and Peters [20] adapt FQI through a soft-greedy action selection.

Riedmiller [16] propose the Neural Fitted Q–Iteration (NFQ), by combining FQI with the neural networks approximator. The author uses the Rprop [25] algorithm for the supervised learning, since it is either fast or insensitive to the choice of the learning parameters. However, results on convergence are still unproved.

Timmer and Riedmiller [26] combine the FQI algorithm and the CMAC function in order to achieve a new powerful approximator to solve the problem of a relatively small set of sampled state transitions.

2.2 Functional networks

Functional Networks (FNs) are generalized ANNs with new features, such as a good computational power and estimation of parameters via linear systems of equations. Due to the presence of neural functions, the global minimum is obtained in few steps and learning does not require a large data set.

The described benefits are confirmed by various applications in different fields. For instance, Elsebakhy apply successfully FNs to predict the software development effort [35].

Iglesias et al. [36] show that FNs provide better results than ANNs as for the prediction of catches of some fish species, with data proceeding from remote sensors. Bruen et al. [37] use FNs to forecast cases of real-time flood. In particular, through a suitable comparison between results achieved by FNs and ANNs, they analyze the ability of FNs to predict flows for short lead-times with less computational cost. Tomasiello [38] uses FNs to foresee fresh and hardened properties of self–compacting concretes, proving that FNs give good results with a low computational cost even when a small data set is used. In this case, obvious comparisons are also done for ANNs.

Rajasekaran et al. [39] consider the approach by Castillo et al. [40], which combines FNs and finite differences to predict the tidal level. Acampora et al. [34] propose an extended version of this approach for a gas sensing system.

Hybrid schemes also refer to FNs with the aim of improving support vector machines and fuzzy logic systems [41]. Sanchez-Marono and Alonso-Betanzos [27] analyze a scheme, that combines Analysis of Variance and FNs, in order to obtain a more efficient feature extraction, as shown by some binary classification problems. Pruneda et al. [28], Zhou et al. [29] and Lacruz et al. [30] discuss some possible applications of FNs in two-class classification. In particular, Lacruz et al. [30] prove that, when the logistic function is adopted, logistic regression techniques are equivalent to the classification procedure by FNs.

FNs are also used in some multi–class classification problems, i.e. to classify computer intrusions [31], as well as thalassemic patients [32] or rock types [33]. A generalized version of FNs for a gas classification problem is proposed in [42].

3 The proposed approach

In what follows, we provide a general overview of the proposed approach. Precisely, the idea of the FQI is combined with FNs, giving a computational scheme named FQI–FN.

First, the Fitted Q-Iteration (FQI) and Functional Networks (FNs) are described. Then, the FQI-FN scheme is presented.

3.1 Fitted Q–iteration

Consider a system with discrete–time dynamics, described by a state space X and an action space U, defined through:

$$x_{t+1} = f\left(x_t, u_t\right), \quad t = 0, 1, \ldots$$

being t the discrete time step and $x_t \in X$, $u_t \in U$. Assume that to a transition from t to $t+1$ is associated an instantaneous reward signal $r_t = r\left(x_t, u_t\right)$, being r a bounded reward function. The actions are selected according to a policy $\pi : x \in X \rightarrow \pi(.|x) \in P(U)$, representing $P(U)$ the transition probabilities.

A way to choose the first action to be applied is provided by the state-action value (or Q-) function. It is defined as the expected cumulative reward starting in a state x, taking an action u and following the policy π:

$$Q^{\pi}(x,u) = E^{\pi}[\rho_t | x_t = x, u_t = u] = E^{\pi}[\sum_{k=0}^{\infty} \gamma^k r_{t+k} | x_t = x, u_t = u] \tag{1}$$

where $0 \leq \gamma \leq 1$ is a discount factor that weights short-term rewards more than long-term ones.

The optimal state-action value function Q^* satisfies the Bellman equation:

$$Q^*(x,u) = E'[r_t + \gamma \max_{u \in U} Q^*(x_{t+1}, u_{t+1})] \tag{2}$$

denoting E' the expected value.

The Q-value iteration is a recursive method for computing an optimal policy and its value. It starts from an arbitrary value, by refining an estimate of Q^*.

For high dimensional states, the Q-value iteration is not feasible in practice, and it is better working with an approximation of the Q-function.

This consideration motivated the Fitted Q-Iteration.

Fix a temporal horizon of N steps, and indicate by:

$$\pi_N\left(t, x\right) \in U,\; t \in \left\{0, ..., N-1\right\},\; x \in X$$

a N-step control policy (i.e., $u_t = \pi_N(t, x_t)$), and by:

$$J_N^{\pi_N}(x) = \underset{t=0,1,\ldots,N-1}{E'}\left[\sum_{t=0}^{N-1} \gamma\, r(x_t, \pi_N(t,x))\,|x_0 = x\right]$$

the expected return of $\pi_N(t,x)$ over N steps. An N–step optimal policy π_N^* is a (not necessarily unique) policy that maximizes J_N^{π} for any x.

The FQI algorithm gives a sequence of Q_N functions defined on $X \times U$ by

$$Q_0(x,u) = 0 \tag{3}$$
$$Q_N(x,u) = (HQ_{N-1})(x,u), \forall N > 0 \tag{4}$$

where H is an operator mapping any function $K : X \times U \to \Re$. It is possible to show that the sequence of Q_N functions converges to the solution of the Bellman equation 2 (an exaustive presentation is in [15]).

The sequences:

$$\pi_N^*(0,x) = \arg\max_{u' \in U} Q_N(x,u'), \forall\ N > 0,$$

$$\pi_N^*(t,x) = \pi_{N-1}^*(t-1,x), \forall\ N > 1,\ t \in \{0,\ldots,N-2\}$$

are N–step optimal policies with expected returns over N steps given by:

$$J_N^{\pi_N^*}(x) = \max_{u \in U} Q_N(x,u).$$

The sequences:

$$\pi_N^*(0,x) = \arg\max_{u' \in U} Q_N(x,u'), \forall\ N > 0,$$

$$\pi_N^*(t,x) = \pi_{N-1}^*(t-1,x), \forall\ N > 1,\ t \in \{0,\ldots,N-2\}$$

are N–step optimal policies with expected returns over N steps given by:

$$J_N^{\pi_N^*}(x) = \max_{u \in U} Q_N(x,u).$$

3.2 Functional Networks

A FN is a structure that consists of the following elements:

- *a layer of input units* for the input signals;
- *a layer of output units* for the output data;
- *one or more layers of inner neurons*, useful to evaluate the input signals from the previous layers and provide the output signals to the next layer by appropriate functions.

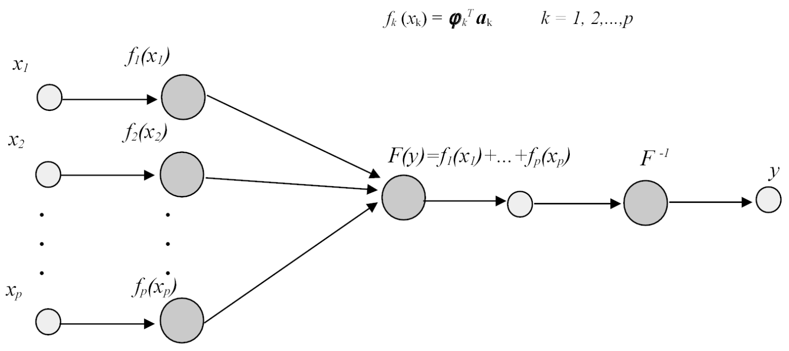
$f_k(x_k) = \boldsymbol{\varphi}_k^T \boldsymbol{a}_k$ $k = 1, 2, \ldots, p$
x_1
x_2
x_p
$f_1(x_1)$
$f_2(x_2)$
$f_p(x_p)$
$F(y)=f_1(x_1)+\ldots+f_p(x_p)$
F^{-1}
y

Figure 1: A Functional Network

Figure 1 shows an additive functional network with p input variables $x_1, x_2, \ldots, x_p$ and an output variable y. The various functions $f_1, f_2, \ldots, f_p$ are approximated through a basis ϕ_k, which satisfies the relation:

$$f_k(x_k) = \phi_k^T \mathbf{a}_k, \qquad k = 1, 2, \ldots, p, \tag{5}$$

where: $\mathbf{a}_k$ is a vector of $m+1$ unknown coefficients, where m represents the chosen approximation order; ϕ_k is usually chosen considering some function groups, such as:

- a polynomial family, i.e. $\phi_k^T = (1,\, x_k,\, x_k^2, \ldots,\, x_k^m)$;
- a cosine family, namely $\phi_k^T = (1,\, \cos x_k,\, \cos 2x_k, \ldots,\, \cos mx_k)$.

Possible hybrid choices for ϕ_k are simply obtained via combinations of different function families.

From relation (5), the output y is computed as:

$$F(y) = \sum_{i=1}^{p} f_i(x_i),$$

where a suitable linear approximation is used.

The Lagrange multipliers method, solved using standard algorithms ([43]), is useful to get the unknown coefficients and ensure the uniqueness of the solution. This method is not suitable when $F(y)$ is not invertible, hence a nonlinear least-square algorithm is needed. Indeed, in this case the existence of a single optimal value is not guaranteed, so a minmax method is usually used, see [44], to overcome this drawback. The idea of the method is to minimize the maximum absolute error between predicted and observed values. The learning method leads to a linear programming problem and the global optimum is obtained in a finite number of iterations.

3.3 The FQI–FN scheme

The proposed scheme is constructed assuming F as the identity function with auxiliary condition $\phi_{10}^T \mathbf{a}_1 = \beta$, where $\phi_{10} = \phi_1(x_0)$ and β is a threshold value.

Define the quantities:

- ϕ_0: vector with null elements except the first $m+1$ ones, given by ϕ_{10};
- N_d: number of training samples;
- $\mathbf{x}_j$: vector referred to the $j-$th pattern with elements x_k, $k = 1, \ldots, p$,

and consider the error function in terms of the distances between the approximate and the exact values, $e_j = \phi(\mathbf{x}_j) - y_j$:

$$E = \sum_{j=1}^{N_d} e_j^2 = \mathbf{a}^T \sum_{j=1}^{N_d} \phi(\mathbf{x}_j)\phi(\mathbf{x}_j)^T \mathbf{a} - 2\mathbf{a}^T \sum_{j=1}^{N_d} y_j \phi(\mathbf{x}_j) + \sum_{j=1}^{N_d} y_j^2. \tag{6}$$

Using the Lagrange multipliers method, the following function has to be minimized:

$$S(\mathbf{a}^T) = E + \lambda(\mathbf{a}^T \phi_0 - \beta), \tag{7}$$

where λ is the Lagrange parameter. From the minimization of (7), we get the system:

$$\begin{pmatrix} \mathbf{B} & \phi_0 \\ \phi_0^T & 0 \end{pmatrix} \begin{pmatrix} \mathbf{a} \\ \lambda \end{pmatrix} = \begin{pmatrix} \bar{\mathbf{y}} \\ \beta \end{pmatrix}, \tag{8}$$

with:

$$\mathbf{B} = 2\sum_{j=1}^{N_d} \phi(\mathbf{x}_j)\phi(\mathbf{x}_j)^T, \quad \bar{\mathbf{y}} = 2\sum_{j=1}^{N_d} y_j \phi(\mathbf{x}_j).$$

Using the solution of (8), the output y is computed as:

$$y = \frac{\beta}{c}\phi^T\left(\frac{c}{\beta}\mathbf{B}^{-1}\bar{\mathbf{y}} + \mathbf{B}^{-1}\phi_0 - \mathbf{B}^{-1}\phi_0\phi_0^T\mathbf{B}^{-1}\frac{\bar{\mathbf{y}}}{\beta}\right), \tag{9}$$

where $c = \phi_0^T \mathbf{B}^{-1} \phi_0$ and ϕ is the vector of functions basis.

The FQI–FN algorithm is presented in Figure 2. Assuming that the convergence is guaranteed, the iterative process stops when a maximum number of iterations is reached [15] or when $|Q_N - Q_{N-1}| < \varepsilon$, where ε is a given threshold value. Choosing the identity function for the output and the same parameter β for each iteration, the sequence of Q_N–functions generated by the FQI–FN approach converges if $\left|\frac{\beta}{c}\right| \leq 1$, as proved in [19].

4 Case study

In this section we test the described methodology in two different situations dealing with a shopping mall. Notice that, in spite of the apparent simplicity and the limited number

```
Input
T= {($x_k$, $u_k$, $r_k$, $x'_k$ )|k=1,..., $N_d$}, a basis of admissible functions, $\mathbf{x}_0$ and $\beta$ for the auxiliary condition

Initialize
Set $N$ =0, $\overline{Q}_N$ =0 everywhere on $X \times U$.

Build
- Initial input-output pairs $i_k^0=(x_k,u_k)$, $o_k^0=r_k$
- Matrix $\mathbf{B}$ and vector $\boldsymbol{\varphi}_0$

Iterate
Do until stopping conditions are reached
- $N \leftarrow N+1$
- Induce from the training set the function $\overline{Q}_N$
- Update the  input-output pairs by means of

                    $o_k^{N+1} = r_k + \gamma \max_u \overline{Q}_N (x'_k, u)$
```

Figure 2: Steps of the FQI–FN algorithm.

of actions in interactions between the user and the system, such examples are a quite good compromise to test the potentialities of the proposed approach.

Interactions between the user and the system are dialogue–like and based on a set of actions available to the system in order to provide suggestions and respond to everyone is asking for some information.

4.1 The data set

The generated data consider a shopping mall. First, in order to test the behaviour of the system and focus on a relatively small number of actions N_a, restaurants, as well as clothing, shoes and accessories stores, are analysed. Then, with the aim of increasing the number of actions, a supermarket, a kitchenware shop and an appliance store, are considered.

The system state is represented by the following characteristics:

- position $i \in \{1,\ M_S\}$, that indicates that the user is inside the sector i;
- time, divided into four disjoint intervals, i.e. 9:00-12:00 (1), 12:00-14:30 (2), 14:30-19:00 (3), 19:00-20:30 (4);
- attribute, namely a value that is 1 for stores and 2 for restaurants;
- a boolean flag, that indicates if an attribute has (1) or has not (0) a value.

In order to ensure a sufficient exploration, the generated data are obtained in a random way, neglecting the meaningless cases; as usual, the number of test samples is set as $20\% N_d$ about.

As for the FQI–FN algorithm, for the numerical tests m is chosen as small as possible, β in a random way, $\mathbf{x}_0$ inside the data set, considering the condition $\left|\frac{\beta}{c}\right| \leq 1$ [19].

The results are obtained in terms of a rate $r = 100n_1/n_2$, where n_1 and n_2 are the number of actions, respectively, correctly individuated and with the maximum reward. Finally, a comparison with the results, which are achieved via NFQ with one hidden layer consisting of 50 neurons, is presented.

4.1.1 First case

In the first case, we set $M_S = 5$ with $N_a = 4$, considering $N_d = 20$ and $N_d = 15$.

The possible system actions are the following:

(1) Hi. May I suggest you a restaurant?

(2) Hi. Do you want to know if there is any store nearby with special offers?

(3) Would you say a clothes store?

(4) Would you say a shoes store?

Actions (1) and (2) refer to system initiatives in order to suggest a restaurant in the time intervals (2) and (4) or to indicate special offers in one or more stores in the sector i where the user is located, respectively. Actions (3) and (4) are useful to disambiguate a user request.

For instance, suppose that the user U wants some "brand1", i.e. a brand that mainly produces garments, which then began producing shoes and accessories; a possible interaction with the system S is as follows:

S - Welcome. How may I help you? I know about clothes stores, shoes and accessories stores, restaurants.

U - I would like some brand1.

S - Would you say a clothes store?

U - Yes.

Such example refers to action (3), but it is repeated for action (4) if the user asks for "brand2", i.e. a brand that mainly produces shoes and accessories, which then began producing garments.

Figure 3 shows the rate r, after $N = 9$ iterations, for the two values of N_d for the training and the test. By slightly increasing N_d, the results improve noticeably but are better using FQI–FN (with $m = 5$ and a hybrid basis of functions, that is with polynomial and cosine terms). On the other hand, the test for NFQ fails (0%).

Table 1 presents how the rate r varies when the number of iterations N increases: through FQI–FN, the convergence seems to be faster (for $N > 9$, the results are unchanged). Table 2 shows that the rate r changes for different dimensions of the basis of function m for the hybrid and the cosine family.

4.1.2 Second case

In order to increase the values of M_S and N_a, we added to the actions listed in the previous subsection the following ones:

(5) Would you say a supermarket?

(6) Would you say a restaurant?

(7) Would you say a kitchenware shop?

(8) Would you say an appliance store?

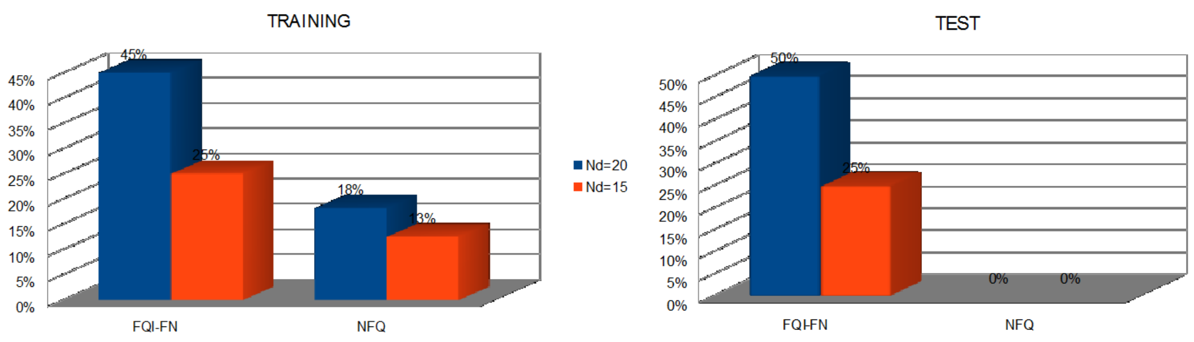

Figure 3: First experiment: the rate r for the training (left) and the test (right).

Table 1: Rate r vs N

Approach	N	r
NFQ	9	18%
NFQ	15	23%
NFQ	25	27%
NFQ	40	27%
$FQI - FN$	5	9%
$FQI - FN$	7	20%
$FQI - FN$	8	27%
$FQI - FN$	9	45%

Table 2: Rate r vs m

Basis	m	r
hybrid	3	36%
hybrid	4	36%
hybrid	5	45%
cosine	3	18%
cosine	4	18%
cosine	5	18%

Actions (5)–(8) are useful to disambiguate a user request. For instance, a user U asks for gluten–free pasta, which is also available in some restaurants, hence the interaction with the system S could be:

U - I would like some gluten–free pasta .

S - Would you say a supermarket?

If the time is inside the interval (2) or (4), S responds as:

S- Would you say a restaurant?

In another example, a user U asks for a "brand0", i.e. a brand which can indicate a frying pan or a household appliance.

We have:

U - I would like a brand0 .

S - Would you say a kitchenware shop?

If the user is in the sector where the appliance store is located, then we get:

S- Would you say an appliance store?

In this experiment, $M_S = 10$, $N_a = 8$, $N_d = 48$ and $N_d = 40$. The behaviour of the solution is similar to the one observed in the first experiment. In particular, the best achieved solution for this experimental session is in Figure 4 that refers to a cosine family with $m = 7$ and $N = 11$.

As for the first experimental session, FQI–FN provides better results that improve by slightly increasing N_d.

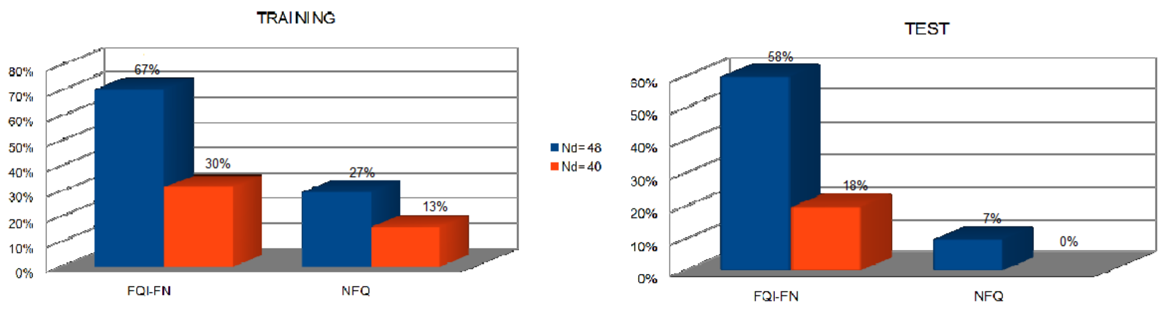

Figure 4: Second experiment: the rate r for the training (left) and the test (right).

5 Conclusions

Considering that ubiquitous recommender systems combine characteristics from ubiquitous and recommendation domains in order to assist the user in some tasks, in this paper a revised FQI algorithm, which uses FNs, is proposed.

The FQI algorithm analyzes the reinforcement learning problems as a sequence of standard supervised learning tasks and FNs are useful as approximators in cases where data sets are small. In particular, FNs operate via a basis of admissible functions that deal with a least squares approach to get the unknown parameters of the approximation.

In order to respond unambiguously to users' initiatives, the approach is used to study a case of items' recommendation in users' proximity and services at a given time during a visit in a shopping mall.

For the experimental results, due to the FNs, a relatively small number of training samples is used. The preliminary presented example shows that the method is promising. Further studies are going to be developed to confirm the goodness of the proposed algorithm.

6 Compliance with Ethical Standards

All the authors declare that they have no conflict of interest.

This article does not contain any studies with human participants or animals performed by any of the authors.

References

[1] P. Resnick, H.R. Varian, *Recommender systems*, Communications of the ACM, 40(3), pp. 56–58 (1997).

[2] A. Gediminas, A. Tuzhilin, *Toward the Next Generation of Recommender System: A Survey of State-of the-Art and Possible Extensions*, IEEE Transactions on Knowledge and Data Engineering, 17(6), pp. 149–156 (2005).

[3] L. Baltrunas, *Exploiting contextual information in recommender systems*, In: Proceedings of the 2008 ACM conference on Recommender systems, pp. 295–298 (2008).

[4] G. Adomavicius, A Tuzhilin, *Context-aware recommender systems*, In: Proceedings of the 2008 ACM conference on Recommender systems, pp. 335–336 (2008).

[5] A. K. Dey, G. D. Abowd, D. Salber, *A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications*, Human-Computer Interaction, 16, pp. 97–166 (2001).

[6] C. Mettouris, G. A. Papadopoulos, *Ubiquitous recommender systems*, Computing, 96(3), pp. 223–257 (2014).

[7] B. Fang, S. Liao, K. Xu, H. Cheng, C. Zhu, H. Chen, *A novel mobile recommender system for indoor shopping*, Expert Systems with Applications, 39(15), pp. 11992–12000 (2012)

[8] A. E. Fano, *Shopper's eye: Using location-based filtering for a shopping agent in the physical world*. In: AGENTS '98 Proceedings of the second international conference on Autonomous agents, pp. 416–421 (1998).

[9] S. Singh, D. Litman, M. Kearns, M. Walker, *Optimizing Dialogue Management with Reinforcement Learning: Experiments with the NJFun System*, Journal of Artificial Intelligence Research, 16, pp. 105–133 (2002).

[10] T. Mahmood, G. Mujtaba, A. Venturini, *Dynamic personalization in conversational recommender systems*, Information Systems and e-Business Management, 12(2), pp. 213–238 (2014).

[11] T. Mahmood, S. H. Ahmed, S. Mahmood, *Comparing Reward-based Optimal Behaviors in User-Adapted Recommender Systems*, In: Proceedings of 3rd IEEE International Conference on Computer Science and Information Technology, 5, pp. 332–336 (2010).

[12] R. S. Sutton, A. G. Barto, *Reinforcement learning: an introduction*, MIT Press, Cambridge, MA, 1998.

[13] C.J.C.H. Watkins, P. Dayan, *Q–learning*, Machine learning, 8, pp. 279–292 (1992).

[14] . N. Taghipour, A. Kardan, *A Hybrid Web Recommender System Based on Q-Learning*, In: SAC '08 Proceedings of the 2008 ACM symposium on Applied computing, pp. 1164–1168 (2008).

[15] D. Ernst, P. Geurts, L. Wehenkel, *Tree-based batch mode reinforcement learning*, Journal of Machine Learning Research, 6, pp. 503—556 (2005).

[16] M. Riedmiller, *Neural fitted Q-iteration – first experiences with a data efficient neural reinforcement learning method*, Machine Learning: ECML 2005, Volume 3720 of the series Lecture Notes in Computer Science, pp 317-328 (2005).

[17] E. Castillo, *Functional networks*, Neural Processing Letters, 7, pp. 151–159 (1998).

[18] E. Castillo, A. Iglesias, R. Ruiz-Cobo, *Functional Equations in Applied Sciences*, Elsevier, The Netherlands, 2005.

[19] M. Gaeta, V. Loia, S. Miranda, S. Tomasiello, *Fitted Q–iteration by Functional Networks for control problems*, Applied Mathematical Modelling, June 2016.

[20] G. Neumann, J. Peters, *Fitted Q–iteration by Advantage Weighted Regression*, Advances in Neural Information Processing Systems, 21, pp. 1177–1184 (2008).

[21] S. B. Nagesh, Z. Lendek, A. A. Khalate, R. Babuska, *Adaptive fuzzy observer and robust controller for a 2–DOF robot arm*, In: Proceedings of IEEE International Conference on Fuzzy Systems, pp. 1–7 (2012).

[22] D. Ormoneit, S. Sen, *Kernel–based reinforcement learning*, Machine Learning, 49, pp. 161–178 (2002).

[23] G. J. Gordon. Online fitted reinforcement learning, in VFA workshop at ML-95, 1995.

[24] A. Antos, R. Munos, and C. Szepesvari, Fitted Q-iteration in continuous action-space MDPs, Advances in Neural Information Processing Systems 20 (2008) 9—16 .

[25] M. Riedmiller and H. Braun, A direct adaptive method for faster backpropagation learning: The RPROP algorithmn Proceedings of the IEEE International Conference on Neural Networks (ICNN), 1993.

[26] S. Timmer and M. Riedmiller, Fitted Q–iteration with CMACs, in Proceedings of IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning (ADPRL), 2007.

[27] N. Sanchez-Marono and A. Alonso-Betanzos, Feature selection based on sensitivity analysis, in Lecture Notes in Computer Science - 12th Conf Spanish Association for Artificial Intelligence and its associated Conference on Technology Transfer on Artificial Intelligence (CAEPIA/TTIA) 4788 (2007) 239–248.

[28] R. E. Pruneda, B. Lacruz, C. Solares, A First Approach to Solve Classification Problems Based on Functional Networks,Artificial Neural Networks: Formal Models and Their Applications, in Lecture Notes in Computer Science - ICANN, 3697 (2005) 313–318.

[29] Y. Zhou, D.-Xu He, Z. Nong, Application of Functional Networks to Solving Classification Problems, World Academy Science, Engineering, Technology 12 (2005) 71–74.

[30] B. Lacruz, A. Perez-Palomares, R. E. Pruneda, Functional Networks for Classification and Regression Problems, in Proceedings of International Conference on Mathematical and Statistical Modeling in Honor of Enrique Castillo, 2006.

[31] A. Alonso-Betanzos, N. Sanchez-Marono, F.M. Carballal-Fortes,J. Suarez-Romero, B. Perez-Sanchez, Classification of computer intrusions using functional networks. A comparative study, in Proceedings of European Symposium on Artificial Neural Networks, 2007.

[32] E. A. El-Sebakhy, A. S. Hadi, K. A. Faisal, Iterative Least Squares Functional Networks Classifier, IEEE Transaction on Neural Networks 18(3) (2007) 844–850.

[33] V. K. David and S. Rajasekaran. Pattern Recognition Using Neural and Functional Networks. Studies in Computational Intelligence, 160. Berlin Heidelberg: Springer-Verlag; 2009.

[34] G. Acampora, M. Gaeta, S. Tomasiello, An extended functional network model and its application for a gas sensing system, Soft Computing, 17(5) (2013) 897–908.

[35] E. A. El-Sebakhy, New computational intelligence paradigm for estimating the software project effort, in Proceedings of International Conference on Advanced Information Networking and Applications (AINA), 2008.

[36] A. Iglesias, B. Arcay, J. M. Cotos, J. A. Taboada, C. Dafonte, A comparison between functional networks and artificial neural networks for the prediction of fishing catches, Neural Computing Applications, 13 (2004) 24—31.

[37] M. Bruen, J. Yang, Functional networks in real–time flood forecasting: a novel application, Advances in Water Resources, 28 (2005) 899—909.

[38] S. Tomasiello, A functional network to predict fresh and hardened properties of self-compacting concretes, International Journal of Numerical Methods in Biomedical Engineering, 27(6) (2011) 840—847.

[39] S. Rajasekaran, K. Thiruvenkatasamy, T.–L. Lee, *Tidal level forecasting using functional and sequential learning neural networks*, Applied Mathematical Modelling, 30, pp. 85—103 (2006).

[40] E. Castillo, A. Cobo, J. M. Gutiérrez, E. Pruneda, *Working with differential, functional and difference equations using functional networks*, Applied Mathematical Modelling, 23(2), pp. 89—107 (1998).

[41] T. Helmy, A. Fatai, *Hybrid computational intelligence models for porosity and permeability prediction of petroleum reservoirs*, International Journal of Computational Intelligence Applications, 9(4), pp. 313–337 (2010).

[42] M. Gaeta, V. Loia, S. Tomasiello, *A Generalized Functional Network for a Classifier-Quantifiers Scheme in a Gas-Sensing System*, International Journal of Intelligent Systems, 28(10), pp. 988–1009 (2013).

[43] W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery, *Numerical Recipes in C, The Art of Scientific Computing, Second Edition*, Cambridge University Press, New York, 1992.

[44] E. Castillo, J. M. Gutiérrez, A. Cobo, C. Castillo, *A minimax method for learning functional networks*, Neural Processing Letters, 11(1), pp. 39–49 (2000).