AI-Based Mobile Context-Aware Recommender Systems from an Information Management Perspective: Progress and Directions

María del Carmen Rodríguez-Hernández*

Aragon Institute of Technology (ITAINNOVA), María de Luna 7–8, 50018, Zaragoza, Spain

Sergio Ilarri*

I3A, University of Zaragoza, María de Luna 1, 50018, Zaragoza, Spain

Abstract

In the Artificial Intelligence (AI) field, and particularly within the area of Machine Learning (ML), recommender systems have attracted significant research attention. These systems attempt to alleviate the increasing information overload that users can experience in the current Big Data era, by providing personalized recommendations of items that they may find relevant. Besides, given the importance of mobile computing, these systems have evolved to consider also the dynamic context of the mobile users (location, time, weather conditions, etc.) to offer them more appropriate suggestions and information while on the move.

In this paper, we provide an extensive survey of recent advances towards intelligent mobile Context-Aware Recommender Systems (mobile CARS) from an information management perspective, with an emphasis on mobile computing and AI techniques, along with an analysis of existing research gaps and future research directions. We focus on approaches that go beyond just considering the location of the user and exploit also other context information. In this study, we have identified that deep learning approaches are promising artificial
intelligence models for mobile CARS. Additionally, in a near future, we expect a higher prominence of push-based recommendation solutions where at least part of the recommendation engine could be executed in the mobile devices, which could share data and tasks in a distributed way.

**Keywords:**  Context-Aware Recommender Systems, mobile computing, context-aware computing, personalization, information management

### 1. Introduction

Nowadays, the huge amount of information available may easily overwhelm users when they need to take a decision that involves choosing among a large set of options. For example, at the time of writing, a tourist who is visiting Madrid (Spain) could find more than 250 tourism apps for his/her mobile device when searching in Google Play [1]. **Recommender Systems (RS)** [2], that offer relevant items (articles, products, objects, or places) to the users, have been proposed as a potential solution to this problem. The main goal of these systems is to recommend certain items based on the (usually implicit) user preferences.

In the last decade of the 20th century, the use of these systems has increased in different application scenarios [3]. Recommender systems have been proposed to suggest a whole range of items, including books, music, movies, news, touristic destinations, friends in social networks, and others [2, 4, 5, 6]. They are particularly popular in e-commerce [7, 8], as providing relevant recommendations to customers can help to improve their satisfaction and increase product sales; indeed, most major companies use RS within their services: for example, we can cite eBay [9, 10], Facebook [11, 12], Netflix [13, 14], Amazon [15, 16], Spotify [17, 18], and Pandora [19], among others. On the other hand, they can also provide value-added to end users. So, in the new era of Big Data [20], given the continuous increase of the volume of information to which users are exposed, recommender systems are a very useful tool, able to learn from the behavior of users and discover their preferences.

Moreover, RS can be considered a key Artificial Intelligence (AI) and Infor-
Information Management (IM) asset (see Figure 1) that can bring benefits for both end users and companies. Indeed, according to a survey by 451 Research [21], predictive analytics and recommendations are the two most important types of Machine Learning (ML) technologies desired by current organizations. In the AI area, the term recommender agent has been used as a general term to refer to personalized search engines, intelligent software agents and recommender systems which assist users who need help to find relevant information [22, 23].

At an abstract level, recommender systems try to predict whether a given item will be appreciated by the user as relevant or not, and for this purpose a variety of AI techniques can be applied (classification, clustering, deep learning, regression, etc.). In this paper, we focus on recommender systems which are considered relevant for intelligent systems and use ML techniques to predict which items should be suggested to a user [24], analyzing those systems from an information management perspective (as systems that can help to reduce the information overload of users) and dealing with topics such as those observed in Figure 2. In the figure, we include recommender systems within the
global area of information systems and information retrieval, showing a significant overlap with artificial intelligence (machine learning and data mining), and also emphasizing the relation of some types of recommender systems with areas such as mobile computing and context-aware computing. The figure provides an overview of related fields, without detailing some subareas of the different fields; for example, within the area of information systems, cognitive information systems, such as those based on the idea of cognitive resonance, where hypothesis can be amplified or weakened during a semantic data analysis, could be mentioned [25]. Given its focus on machine learning techniques, this survey excludes papers on mobile CARS that exploit only pure statistical solutions (e.g., [26]), ontologies (e.g., [27]), or techniques from other fields such as spatial databases (e.g., [28]).

Moreover, our study is centered on recommender systems that are both context-aware and mobile:

- **Exploitation of context data.** Traditional recommender systems deal with applications having only two dimensions, users and items \( (User \times Item) \), and do not consider contextual information (e.g., the location of the user, the time of the day, the day of the week, etc.) during the recommendation process. However, recent approaches have highlighted the importance of
considering the context of the situation in which the recommendation process takes place, to offer more relevant and precise recommendations [29]. As a consequence, the integration of recommender systems and context-aware computing has given rise to the so-called Context-Aware Recommender Systems (CARS) [30] [31] [32].

• Use in mobile scenarios. As the context of a user in a mobile computing scenario is highly-dynamic (e.g., the location of the user and his/her surroundings usually change constantly), recommendation algorithms should be able to effectively and efficiently exploit the dynamic context of the user to offer him/her suitable recommendations and keep them up-to-date. Hence, the combination of context-aware recommendations and mobile computing gives rise to the emergence of mobile Context-Aware Recommender Systems (mobile CARS) [33]. A particular case are the so-called Location-Aware Recommender Systems (LARS) [34], that consider only the context variable location.

How different authors integrate the contextual dimension into the traditional recommendation process for different application domains is explained in several surveys on CARS [35] [36] [37] [38] [39]. There are also a few studies that focus specifically on the mobile CARS field [33] [40] [41], but most of these surveys are more than 6 years old [33] [40], and therefore they do not consider many recent relevant works that have been developed in the field. In addition, in [40] the authors have considered only CARS for vehicular ad hoc networks (VANETs) [42]. As far as we know, the most recent systematic review that provides an overview of CARS for mobile scenarios was presented in [41]. However, unlike our study, the authors do not focus on mobile CARS that apply AI techniques during the recommendation process, and they emphasize the aspects related to the context-aware field rather than those that are relevant in the mobile computing field; in fact, one of the main goals of that study was to identify and classify contextual information in categories, such as the location, social data, time, activity, and the multi-dimensional context. Instead, we propose a framework
to analyze research in the field with an emphasis on mobile computing and AI techniques. Thus, our survey provides a relevant and complementary view over previous studies.

Due to the importance of CARS within the AI and IM areas, in this paper, we analyze and classify the most relevant literature of mobile CARS during the last 10 years (see Figure 3 where key references are highlighted), considering approaches that take into account several context variables. Figure 3 shows that some machine learning models have traditionally been exploited over the years, such as clustering techniques, while there are others whose use has become less prevalent (e.g., traditional supervised learning models). In recent years, these traditional techniques have started to be replaced by some advanced deep learning alternatives (e.g., Convolutional Neural Networks, Recurrent Neural Networks, etc.), which are currently being explored in the mobile CARS field.

The structure of the rest of this paper is as follows. In Section 2, we review the technological context of this survey. In Section 3, we describe the evolution of recommender systems from traditional recommender systems to mobile CARS. In Section 4, we provide an in-depth analysis of techniques applied for mobile context-aware recommendation approaches as well as examples of mobile CARS for specific use case scenarios. Finally, we conclude the paper with some open issues in Section 5.

2. Technological Context

In this section, we introduce the technological context needed to facilitate the understanding of the problem of context-aware recommendations in mobile environments. First, in Section 2.1 we present the basics of mobile computing. Then, in Section 2.2 we focus on the role of sensors and present examples of applications that use sensors of mobile devices in dynamic environments. Finally, in Section 2.3 we describe the main features of context-aware computing as a specific paradigm within the mobile computing environment.
Figure 3: Milestones in the development of mobile CARS that exploit artificial intelligence techniques.

2.1. Mobile Computing

The emergence of portable devices (e.g., smartphones, portable computers, tablets, smartwatches, etc.) and advances in wireless networking technologies gave rise to a new paradigm of computing, called mobile computing. In mobile computing, users with portable devices have access to a shared infrastructure independent of their physical location [43]. This provides flexible communication between people, as well as continuous access to data and network services anywhere and at anytime.

In Figure 4 we show an overview of a mobile computing scenario, where we can see that there are alternatives for long-range communications (e.g., 3G, 4G, and 5G) [44, 45, 46], that require a wide-area infrastructure, and short-range communications (e.g., Wi-Fi and Bluetooth) [47, 48]. A mobile environment
infrastructure, represented in Figure 4, is composed by portable devices and base stations, which serve all the mobile devices within their coverage area or cell, by using wireless communications. The communication among base stations is wired; thus, base stations allow the communication between mobile devices and hosts of the fixed network. Moreover, mobile devices can directly interact without any supporting infrastructure through ad hoc P2P (peer-to-peer) interactions, by using technologies such as Wi-Fi or Bluetooth. In public places (e.g., coffee shops, hotels, airports, libraries, schools, supermarkets, etc.), there are hotspots that offer Internet access to the mobile devices, typically using Wi-Fi technology.

Figure 4: Overview of a mobile computing scenario.

2.2. Sensors

A sensor is a device that converts a physical phenomenon of the environment into an electrical signal [49, 50]. According to the way the data is captured, sensors can be classified into the following types [51, 52]:

- **Physical or hardware sensors**: they provide certain raw data captured from the environment.

- **Virtual or software sensors**: they provide higher-level observations usually obtained by fusing the measurements of several sensors (e.g., a more
precise location can be obtained by combining different positioning mechanisms) [53].

- **Social sensors**: they provide data based on social media, such as data posted in social networks (e.g., Facebook, Foursquare, and Flickr), blogs, or microblogs (e.g., Twitter) [54]; as an example, the proposal in [55] exploits microblogs to detect events in the vicinity.

- **Human sensors**: people can also provide interesting data using their own senses or managing other sensors in specific ways; so, they can provide *volunteered geographic information* (VGI) [56] or participate in *spatial crowdsourcing* [57, 58] tasks.

Users with their mobile devices have become an important source of sensor data, as the sensors available in existing smartphones can be exploited [59, 60, 61]. These include inertial sensors, compasses, GPS receivers, microphones, cameras, proximity sensors, ambient light sensors, accelerometers, gyroscopes, temperature sensors, pressure sensors, and so forth. These sensors have facilitated the development of more flexible and dynamic systems in several domains, such as healthcare [62], social networks [63], environment monitoring [64], and transportation [65, 66]. In Figure 5, we show examples of different types of sensors and contextual variables.

![Figure 5: Overview of different types of sensors and contextual variables.](image)

In recent years, the use of sensors is an essential element for context detec-
We mention below, for different types of context, the mechanisms or sensors typically used to capture the contextual information indicated:

- **Computing context.** It describes hardware (e.g., storage, processing power of the CPU, amount of memory, current CPU and memory usage, battery level), software (e.g., operating system and active applications), and network characteristics (e.g., network connectivity, communication costs, and communication bandwidth) of the mobile device and nearby resources. It is captured implicitly by the device itself.

- **User context.** It describes the user’s environment, including the location of the user, his/her social situation, the user’s interests (or goals), and people nearby. The interests of the user can be obtained explicitly, for example, through a user registration process or by using modules able to capture explicit interest indicators (e.g., a system can identify thematic groups by analyzing social annotations of each user’s preferred resources). Implicit approaches obtain the user’s context information based on interactions of the user with the system.

- **Location context.** It is the spatial location (e.g., latitude and longitude) of a person or object. In outdoor scenarios, it is often sensed by using positioning mechanisms (e.g., GPS) while in indoor scenarios the positioning technologies commonly used are based on short-range signals (e.g., Bluetooth, Wi-Fi and infrared), or by using ZIP codes, trajectory data, and explicit methods that require scanning Radio Frequency Identification (RFID) tags, among others.

- **Social situation context.** It is the current relation between users (e.g., family members, friends, neighbors, co-workers, etc.). For example, this can be information about whether a user is with his/her manager, with a co-worker, or with a friend. The social situation can be explicitly captured from a manual representation of the group structure, or implicitly by
capturing data from a system (e.g., enrollment data from learning management systems [87] or social networks [88]). To obtain indications of the level of collaboration between different members of a group, some systems infer the social relations by analyzing interactions between users [89].

- **Physical context.** It describes the environmental situation concerning the user or system; for example, the amount of lighting, traffic conditions, temperature, weather, and noise levels. It is typically acquired from the environment implicitly (e.g., with a thermometer sensor to determine the temperature of the environment, a light sensor to know if it is day or night, a microphone sensor to measure the noise level, etc.) [90], or captured explicitly by the user [91].

- **Time context.** It can be either entered explicitly by the user (e.g., available study time [91]) or determined implicitly by checking the device’s internal clock (e.g., current time).

- **Activity context.** It is often achieved through mobile phone sensors (e.g., accelerometers, gravity sensors, magnetometers, microphones, and gyroscopes), without interfering with the user’s lifestyle. Using these data, ML techniques can be applied to detect activities, such as the current activity that the user is performing (e.g., walking, running, driving a car, riding a bike, etc.). Some systems require explicit user interactions, such as scanning a QR (Quick Response) code [92] or providing manual text input [87], to obtain information about the activity context.

In Section 2.3, we revisit some concepts of context, determined by using these types of sensors, under the perspective of context-aware computing.

### 2.3. Context-Aware Computing

The interest of exploiting contextual information gave rise to the emergence of (mobile) context-aware computing as a paradigm within mobile computing [93, 94]. Several perspectives on how mobile applications should consider
the context have been presented in the literature [95, 96, 97, 98, 99, 100]. Overall, the main goal of context-aware applications is to examine the user’s context and react to the changes of the dynamic environment to discover information of interest [101].

In [102], the authors define the context as “any information that can be used to characterize the situation of an entity”, where an entity could be “a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves”. Other definitions of context have been introduced in the literature related to the context-aware computing field (e.g., [103, 104]). The meaning of context-aware was defined in [102] by indicating that “a system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user’s task”.

Examples of elements defining the context could be the location, temperature, weather, noise level, activity, traffic conditions, lighting, time of the day, week, season of the year, network connectivity, nearby resources, communication bandwidth, and people accompanying the user, among others. There are certain types of context elements that, according to the circumstances, could be more important than others; for example, if it is raining a person could prefer to stay at home watching a movie rather than to go to run (i.e., the weather element in this case is more important than others). Sometimes authors classify the context by categories; for example, in Section 2.2 we showed some possible context categories. From the perspective of the source and the persistence of information, the context can be divided into two main types [29, 105, 106], which are:

- **Static context**, which does not change frequently. Examples are information regarding the address book, contact list, user profile, user preferences, hardware profile, etc.

- **Dynamic context**, which is highly variable. Examples of dynamic context features are the user’s location, the user’s current task, the closeness to
other people or objects, the weather, the temperature, the speed, the time, the system status, the user’s emotions, etc.

Context awareness represents a generalized model of relevant input data (both implicit and explicit) that allows an application to react to its environment. According to [29], the contextual information can be acquired in several ways, such as the following:

- **Explicit acquisition**: when the user enters contextual information directly into the system (through input fields of an application, by filling out a form or answering specific questions, etc.).

- **Implicit acquisition**: when the context is obtained by observing the user’s behavior, relevant data, and/or the environment (e.g., the user’s location detected by his/her mobile device).

- **Inferred acquisition**: when the system obtains context data using statistical or data mining methods.

Generally, the computing context is acquired implicitly by sensors embedded in mobile devices (see Section 2.2). In [90, 107, 108], surveys on context-aware systems, which highlight the different types of sensors used, are provided.

3. Towards Mobile Context-Aware Recommender Systems

In this section, we describe the evolution of recommender systems. First, in Section 3.1, we present the basics of traditional recommender systems. Then, in Section 3.2, we focus on context-aware recommender systems. Finally, in Section 3.3, we tackle mobile context-aware recommender systems.

3.1. Traditional Recommender Systems

A **Recommender System** (RS) is an application which suggests relevant items (e.g., articles, products, objects, or places) to users [4]. It tries to adapt its
proposals to each user individually, based on his/her preferences. These recommendations can be seen as advice about relevant items that are considered of interest to a particular user. More formally, the task of recommendation can be formulated as indicated in Definition 1.

**Definition 1.** Let $U = \{u_1, u_2, \ldots, u_k\}$ be the set of users and $I = \{i_1, i_2, \ldots, i_t\}$ the set of all possible items that can be recommended. Let $f : U \times I \rightarrow R$ be a utility function that measures how useful item $i$ is for user $u$, where $R$ is a totally ordered set of utility values or ratings (e.g., non-negative integers or real numbers within a certain range). Then, for each $u \in U$ the goal of a recommender system is to find an item $i_u^* \in I$, not yet known to the user, that maximizes the utility function:

$$i_u^* = \text{argmax}_{i \in I} f(u, i)$$

In Figure 6, we show the main elements of a recommender system:

- **The input data** (e.g., the item type requested to the RS and information related to the user profile), which are entered (explicitly or implicitly) by the user to initialize the recommendation process.
- **A database**, which stores information about user and item profiles.
- **The recommendation algorithm**, which uses the input data and the database to suggest a list of items to the user (also known as target user, current user, or active user).

![Figure 6: Simplified overview of a recommendation process.](image-url)
On the one hand, user profiles have information about the characteristics (e.g., age, sex, occupation, country, etc.) and preferences of the users (e.g., a value on a rating scale about an item seen, purchased, or visited). This profile information can be provided explicitly or implicitly by each user. In the explicit case, for example, the user is prompted to manually provide some profile information (e.g., the recommendation system asks the user to select some activities that he/she might like). In the implicit case, the preferences are obtained directly from the user’s interaction with the system, without requiring his/her intervention. On the other hand, item profiles contain the features of items (products, places, or activities) to recommend (e.g., taxis, museums, restaurants, etc.), which are typically characterized by structured attributes (e.g., obtained from a catalog of products or provided by business owners), textual descriptions (e.g., extracted from external sources such as forums), and tags (e.g., generated by a user community), among other types of information that could describe the items. In Figure 7, we show an example of basic information about the relationships between users and items in a restaurant recommender system. In this example, only binary ratings are shown (like or not like).

![Figure 7](image)

Figure 7: Example of user and item profiles in a restaurant recommender system.

One of the fundamental tasks of a recommender system is thus the prediction of a rating: for a particular item not seen by the user, the system should be
able to estimate how the user would evaluate it. If the predicted rating is above a predefined recommendation threshold, then the item can be recommended to the user. A list of suitable items to be recommended to a target user is usually sorted according to the ratings predicted by the system. Depending on how the recommendations are obtained, a recommender system can be classified usually in one of three categories [109]:

- **Collaborative filtering (CF) recommendations** [110][111]. It is the process of filtering information by using techniques involving the collaboration among several users, based on their provided preferences (or ratings about items). Depending on the specific algorithm used, collaborative filtering methods can be classified into the following categories:

  - **Memory-based collaborative filtering** [112]. It is one of the most popular collaborative recommendation techniques and it is based on algorithms to find the k nearest neighbors (kNN). For the prediction of new item ratings, this technique analyzes the entire User × Item matrix of ratings to identify users or items with patterns of similar ratings. In **user-based collaborative filtering (UBCF or user-user collaborative filtering)** [113], the user is recommended items that people with similar tastes and preferences liked. In **item-based collaborative filtering (IBCF or item-item collaborative filtering)** [114], the idea is similar, but based on the similarity between items instead of the similarity between users; in this case, the similarity between items is estimated based only on the ratings they receive (items with similar rating vectors are considered to be similar).

  - **Model-based collaborative filtering** [112]. It applies machine learning or data mining techniques (e.g., Bayesian networks, linear classifiers, clustering, neural networks, association rules, etc.) to learn a model (or common patterns of behavior), by using available interaction information provided by the users to the system (i.e., the ratings provided by the users). The model learned is then used to generate the
predictions about the missing interactions.

- **Content-based recommendation** [115] [116]. It recommends to the user items similar to the ones the user preferred in the past. As opposed to the item-based collaborative filtering approach, the ratings provided by other users are not exploited; instead, the similarity between items is computed by taking into account their descriptions (i.e., their features or attributes). Sometimes, the information about the items is a textual description (or a document), which can be structured and exploited to provide content-based recommendations. In this scenario, text mining techniques, used in the *Information Retrieval (IR)* field [117] [118] [119], play an important role.

- **Hybrid recommendation** [120]. It combines several techniques, such as collaborative filtering and content-based methods. Most of the time, hybrid recommendation algorithms are motivated by the need to increase the quality of the recommendations and minimize the weaknesses of individual methods.

Overall, the use of recommender systems has been successful to alleviate the problem of information overload, increase the number of items sold, encourage sales of the most diverse and novel items, facilitate a better understanding of the user’s needs, and increase the satisfaction and fidelity of the users [4] [109]. However, there are still challenges and constraints that offer research opportunities, related to topics such as:

- The *cold start problem*, which occurs when a user or item is new for the recommender system [121] [122] [123] [124] [125] [126] [127] [128] [129] [130] [131] [132].

- The incorporation of *contextual information* during the recommendation process [29] [33] [36] [37] [109] [134].

- The *scalability* of recommendation algorithms, taking into account large
real-world datasets. An example of a potential solution to this problem is the exploitation of binary codes to represent users and items in a compact way while minimizing the quantization loss, to enable efficient and scalable recommendations.

- The support of multi-criteria ratings. For example, a user can evaluate a restaurant (e.g., on a scale of one to five) regarding different aspects or criteria (e.g., rating_food = 3, rating_decoration = 4, rating_service = 5, and rating_price = 4), rather than using a single criterion rating (e.g., rating = 4) like in traditional recommender systems.

- The privacy-protection between users in RS. Recommender systems must be able to keep the personal information of the users private, including their preferences, as users should not be tracked against their will.

- The design of recommender systems that operate on mobile devices.

- The proactive recommendation of items without the need to generate explicit queries.

- The diversity of items recommended to a target user.

- The serendipity (or novelty and unexpectedness of items) in recommender systems.

- The application of strategies that deal with the sparsity problem that arises because the number of ratings provided by users is usually very small compared to the number of unknown ratings (and consequently the rating matrix is very sparse).

- The use of distributed architectures (e.g., P2P networks) in recommender systems.
• Recommendations to *groups of users* with common interests [189, 190, 191, 192, 193, 194, 195, 196].

• The delivery of *explanations*, that allow the user to know the reasons for the recommendations received. These explanations can be based on ratings of similar users, attributes that describe the items, or the use of conversational systems (e.g., questioning and answering techniques) [197, 198, 199, 200, 201, 202, 203, 204, 205, 206]. Knowledge-based recommender systems can lead to useful explanations, which are often difficult to obtain with pure data-driven approaches based on statistics or subsymbolic AI techniques such as neural networks; as an example, motivated by this, Virtual Bartender [207] proposes a combination of data-driven and knowledge-based recommendations. A survey on the integration of symbolic and subsymbolic techniques for explainable artificial intelligence (XAI) appears in [208].

In-depth studies of some of these challenges can be found in [4, 5, 6].

3.2. Context-Aware Recommender Systems

Most RS operate in a two-dimensional (2D) *User × Item* space. However, with advances in the fields of ubiquitous and mobile computing, the lack of analysis of contextual information in recommender systems has been strongly criticized [29, 109, 209, 210]. So, whereas researchers and developers had previously mainly focused on solving classic problems of recommender systems, such as the *cold start* problem [121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131], *high dimensionality* [211, 212], *spam vulnerability* [213, 214], and many others (see Section 3.1 for other challenges of traditional RS), researchers working on recommender systems have recently recognized the need to investigate them in domains where the contextual information is particularly relevant [32, 215, 216, 217].

For example, considering only information about the users and items is not enough in applications such as the recommendation of vacation packages. In
this case, it is important not only to determine which items should be recommended, but also when these recommendations should be provided and how to combine them in a ranked list. Moreover, traditional collaborative filtering techniques generally take into account all the collected ratings of the items to generate the recommendation models; these techniques assume that the context is homogeneous, but actually a user can assign different ratings to the same item in diverse contexts, as the relevance of a specific item may depend on the current context of the user. Therefore, additional contextual information (e.g., the time of the day, with whom the user is with, the weather conditions, what the user is doing, etc.) should be considered in the recommendation process. Examples like this one have motivated research on Context-Aware Recommender Systems (CARS) [30, 31, 32, 38]. A pioneer proposal for CARS is the one by Adomavicius et al. [30, 31, 32]. To improve the recommendations based on contextual information, they extend the classical 2D paradigm to a multidimensional recommendation model that provides recommendations based on multiple dimensions: User $\times$ Item $\times$ Context. So, besides considering the information of users and items, CARS take into account context information, which is a set of contextual attributes $C = \{c_1, c_2, \ldots, c_q\}$. In particular, those authors introduced three different context-aware recommendation paradigms:

- **Pre-filtering**, where the contextual information helps to filter the data before applying traditional recommendation algorithms.

- **Post-filtering**, where the contextual information is considered only in the final step of the process. So, contextual information is initially ignored and the ratings are predicted using any conventional 2D recommender system, taking all the potential items to recommend into account. Afterwards, the resulting set of recommendations is adjusted (contextualized) for each user by using contextual information.

- **Contextual modeling**, where the contextual information is used directly in the modeling technique as part of the estimation of ratings.
The pre-filtering and post-filtering methods consider the context as an additional filtering step that can be applied to any traditional recommendation algorithm, either to restrict its input (pre-filtering) or its output (post-filtering). On the other hand, contextual modeling recommender systems imply a radically different approach, as the contextual information directly affects the generation of the recommendation models.

In several studies, the pre-filtering, post-filtering and contextual modeling paradigms have been compared [133, 218, 219, 220, 221]. The experimental analysis provided in [133] shows that none of the considered context-aware recommendation paradigms dominates the others, considering their predictive performance and diversity measures. However, the performance was affected by several factors, such as the type of recommendation task (e.g., find all the relevant items or only the top-k items), the granularity of the context information, and the type of dataset (e.g., depending on features such as the existence of a high or low sparsity and the heterogeneity of data).

In certain occasions, the problems to be solved require the combination of several recommendation techniques. Hence, the combination of context-aware recommendation paradigms facilitates the emergence of new proposals. An example is the approach proposed in [30], which uses several pre-filtering models and combines their outputs. Another interesting hybridization could be to combine the pre-filtering and post-filtering paradigms [222]. For example, sometimes using the pre-filtering approach may be more useful for attributes such as the day of the week, while for context attributes like the weather the post-filtering approach might be more appropriate.

Many researchers apply AI techniques in the models they propose for the design of CARS. In Figure 8, we show examples of artificial intelligence techniques applied for the development of context-aware recommendation approaches. We identified the use of these techniques only in the pre-filtering and contextual modeling paradigms. In post-filtering paradigms, traditional recommender systems are exploited and then the resulting item list is filtered or adjusted by using contextual constraints (e.g., filtering the items based on their distance...
from the user, and then adjusting the candidate list to solve conflicts). For example, in a scenario of recommendation of points of interest (POIs), the authors of [223, 224] re-ordered the list of candidate items to recommend in such a way that the distance that the mobile user will need to traverse to access those items (e.g., museums, restaurants, etc.) is minimized. In the following, we will discuss in more detail the three paradigms separately.

3.2.1. Pre-filtering Paradigm

In the pre-filtering approach, the contextual information helps to select the most relevant data (User × Item) for 2D recommendations. In Figure 9, we show the general process of the pre-filtering paradigm. Firstly, the contextual information is used to filter out irrelevant ratings. Then, a traditional recommendation model based on contextualized data suggests appropriate items to the user.

The pre-filtering paradigm is also known as the reduction-based approach, as it reduces the problem of multidimensional contextual recommendations to the traditional 2D recommendation space [30]. For example, in a context-aware music recommender system, if a person enjoys listening to music while running
and is at the moment practicing that activity, then the recommender system will only use rating data \((User \times Item \rightarrow Rating)\) related to the context *running*.

An advantage of this approach is that it supports all the 2D recommendation models proposed in the literature [109]. However, if it discards a large amount of data, then the model may not have enough data to generate reliable recommendations. Moreover, it would be interesting to enhance the pre-filtering paradigm with the incorporation of context hierarchies. For example, the context \(C = \{Girlfriend, Theater, Saturday\}\) could be generalized to \(C' = \{WithCompany, AnyPlace, AnyTime\}\).

### 3.2.2. Post-filtering Paradigm

The basic idea of the post-filtering approach is to consider the context as an additional constraint to verify a posteriori. As shown in Figure 10, this paradigm does not take into account the contextual information in the initial data input of the 2D recommendation model. Only the ranked list of candidate items (obtained by using a traditional 2D recommendation model) will be adjusted with the contextual information. This adjustment can be performed in two ways [29]:

- **Filtering** (or selecting) the most relevant items in a given context. In a context-aware book recommender system, an example of item filtering would be the following: if a person usually reads science books over the weekend, the system may remove non-science books from the candidate list of books to recommend during that period.
Figure 10: Post-filtering paradigm.

• Adjusting the ranking of the list retrieved based on a given context. Following with the same scenario, if a ranking adjustment strategy is applied instead, books with more stars (i.e., better valued) written by the authors preferred by the user in that specific context could have a higher value in the ranked list.

In the filtering adjustment, if there are very few contexts similar to the one of the current user, then many items from the candidate list to recommend could be eliminated (even all the items, in the worst case). In the case of ranking adjustment, if there are no similar contexts, then an approach equivalent to a traditional recommendation would be applied.

Moreover, the contextual post-filtering approaches (for both forms of adjustment) can be classified into the following types [29]:

• Heuristic post-filtering approaches, which try to find the common item features for a user in a given context, and then use these features to adjust the list of recommendations.

• Model-based post-filtering approaches, which build models to predict the probability that the user will prefer an item type in a given context (e.g., likelihood of choosing books of a certain literary genre), and then use this probability to adjust the list of recommendations.

Like in the case of pre-filtering, a relevant advantage of the post-filtering paradigm is the ability to use any traditional recommendation model. In addi-
tion, similarly to the pre-filtering approaches, incorporating the ability to manage context generalization models (context hierarchies) into the post-filtering paradigm would be an interesting enhancement.

3.2.3. Contextual Modeling Paradigm

In the contextual modeling approach, the contextual information is used directly in the recommendation model. For this purpose, multidimensional (MD) predictive models (e.g., a probabilistic model, decision tree, etc.) or heuristics that incorporate a context dimension in the user and item data are applied (see Figure 11). This contextual approach assumes that context attributes are appropriate features to learn a recommendation model.

![Figure 11: Contextual modeling paradigm.](image)

In this paradigm, traditional 2D recommendation algorithms cannot be used directly (unlike in the case of the pre-filtering and post-filtering paradigms). However, these can be modified (or extended) with the purpose of incorporating the context dimension in the rating estimation. For example, a traditional neighborhood-based recommendation approach [225] was extended to the multidimensional case in [30].

3.3. Context-Aware Recommender Systems in Mobile Environments

Some context-aware recommendation architectures have been proposed in the literature [226, 227, 228]. However, these architectures are not designed with mobile users in mind, where the context and the movements of the users may be important factors to consider when deciding which items should be recommended. This problem is indeed now a future research direction.
Moreover, the widespread availability of mobile devices, such as smartphones and portable computers, implies that the relevance of mobile computing scenarios is nowadays undeniable. This, in turn, demands new approaches for the development of recommender systems that can handle and effectively exploit the data available in those environments. Hence, the combination of context-aware recommendations and mobile computing gives rise to the emergence of Mobile Context-Aware Recommender Systems (mobile CARS) [33, 40, 41, 67].

The main goal of mobile context-aware recommender systems is to suggest the right items (or services) to mobile users anywhere and at anytime, being the contextual information a key element in determining the relevance of the items. In mobile environments, where the user is moving and the context is highly dynamic, it is essential to provide precise recommendations and avoid overloading the user with the suggestion of many items. Regarding the specific recommendation method used, during the design of a mobile context-aware recommender system it is necessary to decide a suitable contextual recommendation paradigm (i.e., pre-filtering, post-filtering, or contextual modeling), or a combination of these, which best fits the problem to be solved. Besides, the appropriate way for the user to request or receive the recommendations (pull or push approach) has to be determined. Likewise, during the recommendation process, the answer to a user’s query must be continuously re-evaluated by the system until the user decides to cancel it, as the recommended items can change continuously with context changes. From the perspective of mobile computing, recommender systems are characterized by the following elements [33]:

1. **User mobility**: the users can access a mobile information system in different locations, while moving.
2. **Device portability**: the device used to access the information system is mobile (e.g., a smartphone, a tablet, a portable computer, etc.).
3. **Wireless connectivity**: the device used to access the recommender system uses wireless communication technologies (e.g., Wi-Fi or Bluetooth).

In [229], the authors identified three important factors that can influence
the accuracy of mobile CARS: the context, the recommendation method, and privacy considerations. On the one hand, the type of context (e.g., computing context, user context, physical context, etc.) to be included in the recommender system must be determined considering the target recommendation domain. On the other hand, traditional CARS include static context information (e.g., gender, age, contact list, etc.), generally provided explicitly by the users. However, in mobile scenarios, the context is highly changing. For example, in a taxi recommendation scenario both users and items to recommend can be on the move. Hence, an important aspect to consider in this type of recommender systems is the exploitation of dynamic context information (e.g., the location, transport way, mobility, time of the day, etc.), captured from the environment through sensors (e.g., accelerometers, optical sensors, microphones, etc.) embedded in mobile devices and other available data sources (e.g., social networks, traffic web services, etc.). In addition, mobile CARS must be able to automatically update the contextual information of users and items. An advantage of acquiring dynamic context information implicitly is that the users could avoid entering the information manually into the system. Currently, some mobile CARS use dynamic context information (e.g., mood, companion, etc.), but require the users to enter the information explicitly into the system. The problem is that many users do not usually enter this type of information (as it takes time and it is not convenient for them), and then the system lacks relevant contextual information needed to generate accurate recommendations. As an example, the STS dataset [230] (obtained with the South Tyrol Suggests mobile app) collects information about 14 context attributes, but 89.37% of the actual context values are missing [231].

To overcome the problem mentioned above, context data should be captured automatically, whenever it is possible, by using sensors. In Section 2.2 we emphasized the importance of sensors for the topic covered in this survey from a general perspective. Table 1 shows a summary of some physical sensors used in the literature of mobile CARS for the acquisition of highly-dynamic contextual information, which is of key importance for the particular case of mobile
CARS. For example, the user’s current location (e.g., home, college, campus, classroom, library, etc.) and movement trajectory are context data generally acquired from GPS sensors, which is one of the most commonly used sensors in mobile CARS. Through Wi-Fi, the mobile device can identify other nearby networks, as well as extract information that characterizes them. The noise level around the user (e.g., silent, normal, and loud) can be determined by using the microphone embedded in the user’s device. By using the light detector, the ambient light level (illumination) can be obtained to infer the time of the day (e.g., morning, afternoon, evening, and night), in case a GPS receiver is not available (as otherwise the precise time of the day could be obtained through the GPS receiver). Accelerometer and gyroscope sensors measure the acceleration and rotation rate of the mobile device, respectively; using the values obtained from these sensors, the activity that the user is performing (e.g., walking, running, sitting) or his/her transport way (e.g., on foot, bicycle, car, bus, subway) can be inferred. On the other hand, the temperature is an example of useful contextual information (e.g., to determine if it is hot, normal, or cold) that is usually acquired through thermometer sensors. As a final example, a number of user’s physiological conditions can be measured by using biomedical sensors or wearables (e.g., the stress level of the user can be determined by measuring his/her heart rate).

In Table 2, we show examples of virtual, human and social sensors used to obtain contextual information for mobile CARS. Regarding virtual sensors, some CARS exploit certain functionalities of the mobile device’s operating system to capture relevant contextual information, such as the ringer mode (e.g., sound, vibrate, and silent), battery information (e.g., battery level, battery temperature, battery status), day of the week, time of the day, user activity (e.g., use of calls and SMS), mobile apps that are active, etc. On the other hand, some recommender systems use web services available on the Internet (e.g., external weather forecast services) to extract contextual information, such as the temperature (e.g., hot, normal, cold), humidity (e.g., dry, humid, normal) and weather conditions (e.g., sunny, cloudy, clear sky, rainy, snowing) of the envi-
Work | Physical Sensors
---|---
| GPS | Wi-Fi | Microphone | Accelerometer | Light detector | Gyroscope | Thermometer | Biomedical and ambient sensors |
---|---|---|---|---|---|---|---|
[232] | ✔ | ✔ | ✔ | ✔ | ✗ | ✗ | ✗ |
[233] | ✔ | | | | | | |
[234] | | | | | | | |
[235] | | | | | | | |
[236] | | | | | | | |
[237] | | | | | | | |
[238] | | | | | | | |
[239] | | | | | | | |
Table 1: Examples of physical sensors used by mobile CARS for context acquisition.

environment. In university contexts, web services based on maps (for indoor and outdoor environments) have been used to obtain the locations of users on the campus [244]. Another way of detecting the location of mobile users is through the geolocation API of HTML 5, which abstracts the programmer from the use of specific sensors. An example of human sensors could be represented by physical actions performed by clinicians in operating rooms (e.g., surgical actions such as administering anesthesia, performing an intubation, or performing an incision), which are provided to the system to recommend virtual actions in cases of complications and emergencies. As a final example, social networks are used as sensors to capture relevant information from users, such as check-in records of POIs, as well as information about friends and their preferences.

4. Mobile CARS in Depth

In this section, we analyze mobile CARS in detail, as they are one of the most challenging and representative types of CARS. First, in Section 4.1, we present the main techniques used for mobile CARS, considering both pull-based recommendation approaches and push-based approaches. Then, in Section 4.2
Table 2: Examples of virtual, human and social sensors used by mobile CARS for context acquisition.

4.1. General Approaches for Mobile CARS

We classify mobile context-aware recommendation approaches into two main categories: pull-based approaches and push-based (proactive) approaches. In the first case, we assume that the user actively (or explicitly) requests recommendations. In the second case, the user, under certain contextual conditions, implicitly receives recommendations without explicit user requests. A common characteristic in both cases is that the contextual factors (e.g., location, temperature, transport way, etc.) are dynamic, in the sense that they can change continuously. Tables [5] and [4] present an overview of several approaches for mobile CARS and highlight the AI techniques used in each of them.

4.1.1. Pull-Based Mobile CARS

Pull-based mobile CARS follow a request–response pattern. These systems only recommends items if a user makes an explicit (or query-based) request. Several pull-based context-aware recommendation approaches have been proposed for mobile environments [224].
In Figure 12 we present an overview of pull-based recommendation models proposed in the literature, by taking into account their evolution and performance. There are authors who extend 2D recommender systems to achieve CARS (e.g., [253, 254]; see in Figure 12 the arrows from boxes labeled with “RS” to boxes labeled with “CARS”) while others evolve existing CARS (e.g., [232, 255]; arrows from “CARS” to “CARS” in Figure 12). As the figure shows, the proposed n-dimensional MF-based recommendation models have outperformed some classical MF-based recommenders (e.g., S-DEEPREC [251] outperforms LibFM [256] and SVD++ [257]), as well as other n-dimensional MF-based context-aware recommendation models (e.g., GeoMF [258]). On the other
<table>
<thead>
<tr>
<th>References</th>
<th>Description</th>
<th>AI techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td>[245]</td>
<td>Context-aware recommendation approach that applies a predictive linear model to infer the relevance of a POI in a specific contextual situation.</td>
<td>Predictive linear model.</td>
</tr>
<tr>
<td>[235]</td>
<td>Context-aware recommendation approach for surgeons, that recognizes concurrent actions on the raw data obtained from sensors, by using ML techniques, to suggest relevant virtual actions in operating rooms.</td>
<td>Conditional Random Fields (CRF).</td>
</tr>
<tr>
<td>[235]</td>
<td>Recommendation framework that personalizes multimedia contents, by predicting the latent preferences of user’s contexts through adaptive interfaces in an Ambient Intelligent (AmI) environment.</td>
<td>Generative statistical model (Latent Dirichlet Allocation – LDA) and Expectation-Maximization (EM) algorithm.</td>
</tr>
<tr>
<td>[238]</td>
<td>Location-based social network recommender system that uses the user’s location trajectory, user-shared images, and textual comments, to suggest POIs to mobile users.</td>
<td>Deep Learning – Convolutional Neural Networks (CNNs) – and clustering algorithms.</td>
</tr>
<tr>
<td>[238]</td>
<td>Location-aware recommendation model that exploits feed-forward neural networks to learn the latent factors of users and locations, by using check-in information.</td>
<td>Deep Learning – Feed-Forward Neural Networks (FNNs).</td>
</tr>
<tr>
<td>[251]</td>
<td>Sequential latent context-aware recommendation model that uses RNNs to learn a nonlinear interaction function between users, items, and contexts.</td>
<td>Deep Learning – Recurrent Neural Networks (RNNs).</td>
</tr>
</tbody>
</table>

Table 4: AI techniques used in different approaches for mobile CARS (2/2).

hand, the exploitation of DL-based techniques in CARS has represented a significant improvement regarding several state-of-the-art MF-based recommenders (e.g., [251][252][255]). In the following, we detail the most relevant contributions.

Most contextual modeling approaches proposed in the literature use supervised learning techniques to incorporate contextual information into the recommendation process. The training set of this type of model requires, for each vote of an item, information of each context attribute. Sometimes, contextual information is unknown, and therefore an increase of the number of context attributes could aggravate the data sparsity problem. In [232], the authors emphasized the need to limit the dimensionality of the context representation. Hence, they decided to extend the Context-Aware Matrix Factorization (CAMF) recommendation approach presented in [253] by proposing a Latent Context Matrix Factorization Recommendation (LCMF) approach. The main idea of this new approach is to extract latent context data from a rich set of mobile sensors and use them to improve the recommendation algorithm. To address the
Figure 12: Overview of related work considering the evolution and performance of pull-based recommendation models.

sparsity problem, it performs a selection of the best features (e.g., regarding the location, time, ringer mode, speed, battery, activity, microphone, light, accelerometer, rotation, gyroscope, etc.) that can be used to infer unexplored user contexts, by using deep learning and Principal Component Analysis (PCA). The proposed model was evaluated with an Android application that, based on an explicit query provided by the user, recommends POIs nearby, such as restaurants, bars, entertainment centers, etc. Considering the RMSE (Root Mean Squared Error), the proposed LCMF approach was shown to be superior to the Context-Aware Matrix Factorization model [253] and the BiasSGD Traditional MF model [259].

Models based on Matrix Factorization (MF) are one of the most effective collaborative filtering recommenders [253]. However, in [251] the authors presented a deep learning based recommendation model that significantly outperforms
several state-of-the-art recommendation models based on matrix factorization (e.g., SVD++ [257], libFM [256], and GeoMF [258]). The authors explain an improvement in the Normalized Discounted Cumulative Gain (NDCG), due to the high capacity of neural networks to better detect latent factors related to user preferences and locations. Specifically, they implemented a location-aware recommendation model, called \textit{S-DEEPREC}, that exploits feed-forward neural networks to learn the latent factors of users and locations, by using check-ins of users. A novel aspect to highlight is the incorporation of spatial constraints into latent factors related to locations (e.g., geographically-close locations are given a higher preference, as opposed to the one assigned to geographically-distant locations). To implement time-aware location recommendation models, they proposed, as future work, to apply other types of neural networks, such as Recurrent Neuronal Networks (RNNs). Recently, in [255], the authors applied Convolutional Neural Networks (CNNs) in a time-aware recommender system to study the changes of user preferences over time. The experimental results show improvements over the CAMF [253], NeuMF [260], BPR-Opt [261], TF [262], CHNMF [263], and ConvMF [264] models.

The \textit{sequential latent context-aware model (SLCM)} [252] is another example that uses deep neural networks to address limitations of traditional MF (i.e., use of a fixed linear function to capture the complex structure of user and item interactions, and explicit definition of latent factors) for collaborative filtering recommenders. Specifically, the authors used RNNs to learn a nonlinear function of user, item, and context interactions. In movie and POI recommendation domains, the proposed model improves the recommendation accuracy (MAE, RMSE, and Hit@K) compared to other state-of-the-art context-aware models (e.g., [259] [253] [232]), which are also extensions of the neural network collaborative framework (NCF) [260]. A relevant aspect of this work is that the authors deal with the user’s privacy problem, by using context sequences observed in the system for different users instead of individual context sequences per user; this is one of the few papers that tackle this challenge. As future work, the authors plan to solve “the new user problem” by extending the SLCM model
(for example, studying recommenders based on user groups), and to apply other neural network architectures such as CNNs.

In the same line of collaborative filtering research based on MF, the authors of [265] addressed the sparsity problem using latent behavior patterns learned from implicit contextual features (e.g., the current location, the time of the day, and the day of the week). They proposed two POI recommendation methods: Global Pattern Distribution Model (GPDM) and Personalized Pattern Distribution Model (PPDM). Both methods differ in the way they learn the behavioral pattern distribution. The first method assumes that all the users have a fixed pattern distribution, while PPDM learns a personalized pattern distribution per user. The GPDM and PPDM models were compared with state-of-the-art models for next POI recommendation [266, 267, 268, 269, 270, 271, 272], by using the Foursquare and Gowalla datasets. The proposed models obtained better recall and NDCG.

Moreover, a context-aware Bayesian hybrid recommender system was proposed in [233]. The proposed model combines content-based and collaborative filtering recommendation models. It uses contextual information (e.g., the location, season, day of the week, time of the day, temperature, etc.) obtained from a mobile device, user ratings, and attributes that characterize the items. To improve the prediction accuracy, a Bayesian Network is applied in both recommendation models.

4.1.2. Push-Based Mobile CARS

Generally, mobile devices such as smartphones have some limitations in comparison to traditional mobile or desktop computers; for example, they usually provide restricted input facilities (e.g., lack of a comfortable keyboard, small display sizes, etc.). So, a recommender system can try to relieve the user from having to type or introduce significant information as an input, by using push-based context-aware recommendations rather than pull-based context-aware recommendations. A push-based context-aware recommendation approach automatically delivers recommendations to the mobile user in an appropriate context,
without explicit requests from him/her.

For example, a proactive contextual recommendation approach that pushes suggestions to the mobile user when the current situation (i.e., the context) is considered appropriate, without explicit user requests, was proposed in [273]. The idea is to determine not only which items to recommend, but also when to make a recommendation. Hence, the proposed approach periodically analyzes the current contextual conditions and, if the current context is appropriate, then a second phase is activated to examine the suitable items to suggest. For example, a gas station recommender system can proactively suggest a gas station when the remaining fuel level is low and a gas station is nearby, without causing much (or any) detour.

From the perspective of AI techniques, a contextual recommendation approach for mobile environments was proposed in [234]. This approach for smartphone users automatically recommends services (or actions related to the volume adjustment, call settings, profile, applications, etc.) in a specific contextual situation. For example, when the user is in a library, the recommendation model activates the vibrating mode automatically and also offers services like book search. Contextual information (e.g., the day, time, location, temperature, etc.) is captured by the available sensors (e.g., accelerometer, temperature, humidity, etc.), and obtained from data stored by applications in electronic calendars, address books, task lists, etc. In this contextual recommendation model, the context values captured by the sensors are represented as fuzzy values to define the context situations, the actions to execute under the current context conditions are determined by using rules, and Bayesian Network techniques are used to classify the incoming calls (e.g., into high-priority calls, low-priority calls, and unknown calls).

An agent-based architecture for context-aware recommendation was introduced in [244]. The eAgora? application is an implementation of the proposed architecture in a university scenario, which is characterized by an environment which is dynamic (there are mobile users), heterogeneous (there are several mobile devices, social interactions, and services available), intelligent (when the
system is able to react smartly to environment changes), and contextualized (when it is context-aware). The application learns the user preferences continuously, by using software agents, to adjust automatically to the context changes and proactively recommend events occurring in the campus (academic events and cultural events).

A context-aware recommender system that pushes information about different types of items (e.g., restaurants, gas stations, attractions, etc.) in an environment of Internet of Things (IoT) was proposed in [250]. It takes into account the user’s contextual information (e.g., the time, outside temperature, human temperature, blood pressure, if the user is out of his/her country, if he/she is on holiday, the oil level in the car, if the car engine is working at the moment, and if it is lunchtime or not) during the recommendation process. When a type of recommendation (e.g., about a hospital, gas station, restaurant, or cinema) is triggered, then a Naïve Bayes classifier is used to provide information of interest to the user. For example, if the system detects that the user’s blood pressure is high, it would recommend hospitals near his/her location.

Most push-based mobile CARS assume the availability of a centralized server that stores a large database about all the ratings that are released over time (e.g., [241, 243, 274, 275]). In [276], the authors analyzed the possibility of using pure mobile P2P networks to exchange relevant data in contexts where no centralized database or server exists. They implemented a simulation application that allows testing a trajectory-based mobile CARS able to proactively push relevant items to mobile users. The idea is to recommend to the user a trajectory to follow within a museum, taking into account the sequence of works of art to observe during the time available. In this scenario, the museum visitors propagate partial amounts of rating data opportunistically (i.e., when they meet each other in the physical space). The experimental results show that the mobile P2P-based recommendation approach shows a performance close to a centralized strategy but it does not outperform it. This is because the mobile P2P recommendation variant to learn the models only uses information collected opportunistically in the local database of the user’s mobile device, rather than
using a centralized server that contains all the information available. However, mobile P2P architectures have potential advantages over centralized solutions, such as the following: they do not imply costs to deploy a required support infrastructure; the mobile users do not incur any cost derived from the use of cellular communications (e.g., 3G, 4G, or 5G) when providing rating information; and they may provide better privacy guarantees, as no centralized server collects all the information provided by the users.

4.2. Examples of Mobile CARS in Different Domains

In this section, we present several examples of context-aware recommender systems for mobile environments in different domains (e.g., recommendation of restaurants, POIs, music, etc.). As a summary, in Table 5 we provide an overview of the context variables and example application domains of existing work for mobile CARS. Moreover, from the perspective of Information Systems (IS), in addition to contextual attributes, mobile CARS take into account additional information related to users and items. In Tables 6 and 7, we present some application examples that use item and user attributes.

Regarding the evaluation of mobile CARS, some works apply rating prediction measures used to evaluate traditional recommender systems. These metrics determine the accuracy of the recommendations taking into account their error. We identified that, among these metrics, the Mean Squared Error (MAE) and Root Mean Squared Error (RMSE) measures are the ones mostly used by researchers in the mobile CARS field. In Table 8 we show some examples of mobile context-aware recommendation proposals that use these metrics. However, mobile CARS are a new generation of recommender systems, that probably require metrics adapted to dynamic environments and context-enriched data sets. On the other hand, some researchers apply Information Retrieval (IR) metrics for the evaluation of mobile CARS. Examples of these metrics are the precision, recall, F-measure, and Mean Average Precision (MAP). Finally, there are ranking metrics focused on the evaluation of top-k recommendations, where the utility of a recommended item is proportional to its position in the ordered list.
<table>
<thead>
<tr>
<th>Reference</th>
<th>Approach</th>
<th>Context variables considered</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>244</td>
<td>Pull</td>
<td>location, time, interest, satisfaction level</td>
<td>POIs</td>
</tr>
<tr>
<td>245</td>
<td>Pull</td>
<td>distance, temperature, weather, season, companion, time of the day, day of the week, crowdedness, familiarity, mood, budget, travel length, transport way, travel goal</td>
<td>POIs</td>
</tr>
<tr>
<td>246</td>
<td>Pull</td>
<td>time, activity, location</td>
<td>POIs</td>
</tr>
<tr>
<td>247</td>
<td>Pull</td>
<td>location, trajectory of the user’s activity, time</td>
<td>POIs</td>
</tr>
<tr>
<td>248</td>
<td>Pull</td>
<td>location, user’s behavior or actions, POI categories preferred by users</td>
<td>POIs</td>
</tr>
<tr>
<td>249</td>
<td>Pull</td>
<td>location, time of the day, day of the week</td>
<td>POIs</td>
</tr>
<tr>
<td>250</td>
<td>Push</td>
<td>time, location, distance, budget, reachability</td>
<td>restaurants</td>
</tr>
<tr>
<td>251</td>
<td>Push</td>
<td>location, time, activity, companion, status of the mobile device (e.g., flight mode), distance</td>
<td>restaurants</td>
</tr>
<tr>
<td>252</td>
<td>Push</td>
<td>location, transport way, distance, activity</td>
<td>gas stations, restaurants</td>
</tr>
<tr>
<td>253</td>
<td>Push</td>
<td>location, time, transport way, companion, distance, fuel level of the car, detour needed, total length of the route, traffic</td>
<td>gas stations, restaurants</td>
</tr>
<tr>
<td>254</td>
<td>Pull</td>
<td>driving style, road type, landscape, sleepiness, traffic conditions, mood, weather, time of the day</td>
<td>music</td>
</tr>
<tr>
<td>255</td>
<td>Pull</td>
<td>activity, music audio content</td>
<td>multimedia: news, music, movies</td>
</tr>
<tr>
<td>256</td>
<td>Pull / push</td>
<td>location, companion, time of the day, date, emotions, weather, things (e.g., physical components, cellphone)</td>
<td>multimedia: news, music, movies</td>
</tr>
<tr>
<td>257</td>
<td>Push</td>
<td>time of the day, location, weather, user plans, price, noise level, availability of parking, smoking, venues’ business hours</td>
<td>leisure activities</td>
</tr>
<tr>
<td>258</td>
<td>Pull</td>
<td>location, time, companion</td>
<td>food, shopping, health services, POIs (for tourists)</td>
</tr>
<tr>
<td>259</td>
<td>Pull</td>
<td>time, location, day of the week</td>
<td>movies, POIs</td>
</tr>
<tr>
<td>260</td>
<td>Pull</td>
<td>location, time</td>
<td>sessions and exhibitors of an event</td>
</tr>
<tr>
<td>261</td>
<td>Pull</td>
<td>location</td>
<td>products of a shop</td>
</tr>
<tr>
<td>262</td>
<td>Pull</td>
<td>location, car availability, car’s maximum speed, car tank, year of manufacture of the car, car millage counter, car’s CO2 emissions, road congestion</td>
<td>transporters</td>
</tr>
</tbody>
</table>

Table 5: Overview of example domains and context variables considered by existing proposals of mobile CARS.
<table>
<thead>
<tr>
<th>Application</th>
<th>Prototype?</th>
<th>User’s attributes</th>
<th>Item’s attributes</th>
<th>Item type</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LARMU</strong> 247</td>
<td></td>
<td>gender, location, interest, satisfaction level</td>
<td>type of the place, location</td>
<td>POIs</td>
</tr>
<tr>
<td><strong>ReRex</strong> 215</td>
<td>✓</td>
<td>companion, crowdedness, familiarity, mood, budget, travel length, transport way, travel goal</td>
<td>descriptions of POIs</td>
<td></td>
</tr>
<tr>
<td><strong>Labs</strong> 239</td>
<td>✓</td>
<td>age, gender, average expense in credit card transactions per year, places where bank clients have spent their money, payment amount, time and date of the purchase, location</td>
<td>category, name, address, location</td>
<td></td>
</tr>
<tr>
<td><strong>UTtravel</strong> 237</td>
<td>✓</td>
<td>age, gender, employment, behavior (actions or interactions with mobile apps)</td>
<td>descriptions of POIs</td>
<td></td>
</tr>
<tr>
<td><strong>DCAPR</strong> 238</td>
<td></td>
<td>location, activity trajectory, age, gender, education, nationality, textual comments, pictures shared in social networks</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td><strong>SAPRS</strong> 240</td>
<td></td>
<td>location</td>
<td>budget, reachability, location</td>
<td>restaurants</td>
</tr>
<tr>
<td><strong>R²Proactive</strong> 244</td>
<td>✓</td>
<td>activity, companion</td>
<td>name, cuisine type, description, name, average price</td>
<td></td>
</tr>
<tr>
<td><strong>Co-ARS</strong> 240</td>
<td>✓</td>
<td>user’s browsing history (real-time clicking on options to get information about specific restaurants), location, transport way</td>
<td>name, location, rating, cuisine type, address, pictures of the restaurant</td>
<td></td>
</tr>
</tbody>
</table>

Table 6: Overview of user and item attributes considered by existing proposals of mobile CARS (1/2).

of recommendations. Examples of these metrics are the \( NDCG \) and the \( P@K \) metric.
Table 7: Overview of user and item attributes considered by existing proposals of mobile CARS (2/2).

<table>
<thead>
<tr>
<th>Application</th>
<th>Prototype?</th>
<th>User’s attributes</th>
<th>Item’s attributes</th>
<th>Item type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProactiveRS [273]</td>
<td>✓</td>
<td>location, companion</td>
<td>price, location</td>
<td>gas stations, restaurants</td>
</tr>
<tr>
<td>InCarMusic [254]</td>
<td>✓</td>
<td>name, genre preferences, sleepiness, mood</td>
<td>genre</td>
<td>music</td>
</tr>
<tr>
<td>ACACF [242]</td>
<td>✓</td>
<td>user’s activity</td>
<td>music audio content</td>
<td></td>
</tr>
<tr>
<td>RecAm [243]</td>
<td>✓</td>
<td>social network, sex, age, location, companion, emotions</td>
<td>content category</td>
<td>multimedia: news, music, movies</td>
</tr>
<tr>
<td>iS3 [245]</td>
<td>✓</td>
<td>age, gender, frequency of purchase, purchase habits</td>
<td>content, price, location, promotion</td>
<td>products of a shop</td>
</tr>
<tr>
<td>Magitti [247]</td>
<td>✓</td>
<td>user plans (content of emails, calendar, appointments, applications used, web pages and documents viewed, messages), location</td>
<td>price, noise level, availability of parking, smoking, location, venue’s business hours</td>
<td>leisure activities</td>
</tr>
<tr>
<td>EventAware [239]</td>
<td>✓</td>
<td>areas of interest, time, location</td>
<td>name, description, location, session start time, days of the event, contact information</td>
<td>sessions, exhibitors</td>
</tr>
<tr>
<td>Transporter and Customer [240]</td>
<td>✓</td>
<td>name, availability status, location</td>
<td>location, max speed, CO₂ emissions (car tank, car model, car maker, year of manufacture of the car, mileage counter)</td>
<td>transporters</td>
</tr>
</tbody>
</table>

4.2.1. Mobile CARS for the Recommendation of POIs

In the domain of RS, the recommendation of POIs has received a considerable attention [277, 278, 279]. In the same way, this has happened in the specific research area of mobile CARS, characterized by a very dynamic environment and the fact that the user is moving. In [247], the authors emphasize the need to develop recommendation architectures that can be applied to real problems. Be-
sides, they consider that many of the proposed approaches are developed based on the interests of service providers rather than clients. Hence, they proposed a location-based advertisement recommender for mobile users, called LARMU, where the classical collaborative filtering approach is modified to include several context dimensions related to mobile users (e.g., their location, time, interest, and satisfaction level). To understand the needs or interests of the users, they applied data mining techniques (e.g., decision tree-based classification rules and the CART algorithm). The proposed algorithm was evaluated in the context of the recommendation of places for shopping, eating, enjoying, drinking, and learning. Considering the MAE, LARMU was shown to outperform a traditional collaborative filtering model. The experimental results were obtained by using data collected from users, but the proposed architecture was not applied in a real-world mobile scenario. Due to this limitation, the initial motivation of the authors was not completely satisfied. In addition, the sparsity problem of ratings remains as a problem that needs to be addressed.

<table>
<thead>
<tr>
<th>Reference</th>
<th>MAE</th>
<th>RMSE</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
<th>MAP</th>
<th>Survey scores</th>
<th>Hit@K</th>
<th>AVG Rank</th>
<th>NDCG</th>
</tr>
</thead>
<tbody>
<tr>
<td>217</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>218</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>219</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>220</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>221</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>222</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>223</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>224</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>225</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>226</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>227</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>228</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>229</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>230</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>231</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>232</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>233</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>234</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>235</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>236</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>237</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>238</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>239</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>240</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>241</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>242</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>243</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>244</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>245</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>246</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>247</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>248</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>249</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>250</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 8: Examples of evaluation metrics used in mobile CARS.
In [245], motivated by the idea of simplifying the user’s effort to provide ratings of items seen in different contexts, the authors proposed a methodology that identifies the existing quantitative dependency between user ratings and context factors. Through this methodology, users first evaluate whether the context influences (increases or decreases the probability of, or has an impact on) the selection of POIs. Then, a predictive linear model is learned with this information to infer the relevance of a POI in a specific contextual condition. To put it in practice, a mobile context-aware recommendation prototype, called ReRex, was developed, that suggests interesting POIs for tourists, according to contextual conditions (e.g., distance, temperature, weather, season, etc.) that the system must consider during the recommendation process. Another important aspect to highlight is the ability of ReRex to explain the recommendations provided, thanks to the use of a predictive linear model which identifies the most relevant contextual factor for each user’s request. For example, the system could explain the suggestion of visiting a museum by arguing that the climatic conditions are unfavorable to be outdoors. The experimental results show that users prefer context-aware recommendations rather than traditional recommendations. However, the authors propose to improve the explanations provided, due to a low user satisfaction with them.

In the line of developing architectures designed for real-world problems, a mobile CARS based on real banking data (e.g., customer profiles, credit card transactions, etc.) was presented in [236]. The proposed model is used to recommend POIs (e.g., restaurants, stores, cinemas, supermarkets, etc.) to bank customers, by considering the places visited by clients where they used the bank’s credit card. In a first phase, the model applies clustering techniques to the information extracted from customers’ bank accounts in order to generate social clusters, to form groups of clients according to the purchases made with their credit cards, their age, their gender, etc. Once the social context of each user is known, in a second phase, the system filters these social contexts by considering the user’s location to find places closer to the user that requested the recommendation. Finally, in a third phase, the user’s context (inferred
from sensor data) is considered to obtain personalized recommendations. For the evaluation of the architecture, the authors developed a prototype, called Labs. In general, users were shown to be satisfied with the recommended places. However, some users were concerned about privacy issues that could arise in real commercial environments. Moreover, the authors proposed the generation of proactive recommendations as future work.

The UTravel application is another example of context-aware mobile recommender system that applies clustering algorithms to identify profiles of similar users. The architecture is composed of two main modules. The first module captures and generates user profiles. In addition, it exploits the k-means clustering model to identify groups of similar users, by taking into account two types of profiles: the demographic profile (with attributes like the age, gender and employment) and the preference profile (e.g., evaluation of POIs, by considering the quality of services, cost, reachability, waiting time, and overall rating). An important contribution of this work is the methodology applied by the authors to include behavioral information in user profiles at both the individual level and the group level, by analyzing user actions, such as the selection of categories, marking POIs as favorites, clicking on a POI to see its details, etc. The second module of the implemented architecture uses user behavior profiles to decide which POIs (e.g., shops, pubs, museums, etc.) to recommend to the user, by using different item filters, such as a location filter, a behavior filter, and a favorite category filter. As future research, the authors proposed to implement a decentralized version of the application for performance reasons.

As another example, TGSC-PMF is a context-aware probabilistic matrix factorization approach proposed to address the data sparsity problem for the recommendations of POIs, by using information about friends. Specifically, this approach uses contextual information obtained from a Location-Based Social Network (LBSN), where each POI is described using a topic model as well as geographical, social and categorical correlations, to generate a POI preference score, which is then integrated into a probabilistic matrix factorization model.

Recently, with the rise of the Big Data era, Deep Learning (DL) techniques
(e.g., Convolutional Neuronal Networks –CNNs–, Recurrent Neuronal Networks, Attentional Models, etc.) have obtained promising results in several research fields, such as in the Computer Vision and Natural Language Processing (NLP) domains. The use of deep learning techniques in recommender systems has also been an emerging trend \cite{250}. For example, a deep CNN-assisted personalized recommendation framework for mobile wireless network users, named DCAPR, was presented recently \cite{238}. An interesting aspect of DCAPR is its ability to feed on multisource heterogeneous spatiotemporal data of mobile wireless network users and different input features, such as image features, text features, and mobile user trajectories. Information about users (e.g., the trajectory of the user activities, textual comments, pictures shared by them, etc.) is extracted from their social networks. In addition, other types of information are captured from user profiles in social networks (e.g., the age of the user, his/her gender, nationality, and education). Users’ location information is also exploited to identify friends with common interests, through the use of clustering techniques; specifically, the authors assume that people who visit POIs located in the same region have common preferences. The CNN architecture has three layers: a rough layer, an enhanced layer, and an accurate layer. The first layer considers users with common trajectory activities as candidates, which are obtained by comparing the trajectories of mobile users in the social network. Some candidates can be fake-friends, as they can have similar trajectories but different visited POIs and therefore different interests (e.g., users visiting different types of stores in the same mall). In the second layer, the CNN model captures features from images published in a social network, to filter the candidate friends with similar POIs. In the last layer, a CNN classifier is exploited to extract text features from comments posted or articles of interest shared by the user in a social network. Concerning precision, recall, and F-measure, the proposed recommendation model outperforms the baselines considered in the paper. An important conclusion of this paper is that, despite the efforts, there is still a gap between deep learning, mobile wireless communication networks, context-aware recommender systems, and mobile computing. According to that work, another
current challenge in this research field is how to combine multisource data to obtain more accurate recommendations.

Regarding the recommendation of POIs, we would also like to highlight that the tourism scenario has been the most exploited case study to implement and evaluate the effectiveness of mobile CARS [26, 27, 237, 245]. On the one hand, this is because in this type of domain it is easy to appreciate the importance of including contextual information in recommender systems to obtain appropriate recommendations. For example, a travel recommender system that assists tourists during their trip may be influenced by various contextual factors, such as the distance to POIs, the weather, the season, whether the POIs are places in open or closed spaces, etc. LOOKER [20] is an example of mobile recommendation application that takes into account the location and time before suggesting different travel-related services (e.g., food, shopping, health, and POIs) to tourists. On the other hand, tourists generally have limited time available and are constantly on the move (visiting POIs). Hence, from the perspective of mobile computing, the tourism domain is a suitable scenario to show the need to implicitly capture and update dynamic contextual information from the environment.

4.2.2. Mobile CARS for the Recommendation of Restaurants

In the restaurant recommendation domain, several approaches have been proposed in the literature. For example, the Situation-Aware Proactive Recommender System (SAPRS) [246] pushes information about relevant restaurants to the user at the right contextual situation. In the first phase, SAPRS applies fuzzy logic as an inference technique to address the uncertainty of the current situation, and if the situation is appropriate then a collaborative filtering recommendation model is activated in a second phase.

A current challenge for mobile CARS is the proper design of mobile user interfaces. Along these lines, [271] evaluated the impact of proactivity on the user experience in a restaurant recommendation domain, to analyze if users would accept proactive recommendations, how to present the recommended items, and
how to properly notify the users. To answer these questions, the authors evaluated two mobile user interfaces (a widget-based interface and a notification-based interface) for context-aware restaurant recommendation, based on the proactive recommendation model proposed in [273]. The experimental results showed that the widget-based interface was preferred by the users.

A pull-based context-aware recommender system for mobile environments, called Co-ARS, was presented in [240]. This system recommends restaurants by considering contextual information and attributes such as the user’s location, restaurant’s location, user’s preferred transportation mode, network distance between the user’s current location and the final destination, and overall rating of the restaurant. The user’s location is automatically acquired from the GPS sensor embedded in a smartphone. The transportation mode used (e.g., stationary, walking, biking, or driving) is detected by using a Bayesian Network classifier. During the rating prediction process, the proposed recommendation model favors nearby restaurants, by considering the mode of transportation and the distance between the user’s current location and the restaurant’s location. Besides recommending a list of nearby restaurants, for each of them the optimal route and travel mode is suggested, by using Google Maps.

4.2.3. Mobile CARS for the Recommendation of Other Items

To facilitate the consumer shopping process, several other mobile CARS have been implemented. An example is the Intelligent Shopping-aid Sensing System (iS³) [248]. The model incorporated in this system first uses the k-means clustering algorithm to generate customer clusters, by considering the gender, age, and frequency of purchase variables. Then, an association rule mining approach (Apriori algorithm) is applied to each cluster to provide product recommendations to customers. The proposed recommendation system integrates RFID technology to automatically show information about products (e.g., size and specifications) to users.

Other works focus on the user’s activity to generate recommendations. For example, Magitti [241] is a scalable architecture for context-aware activity-
detecting mobile recommender systems. It infers leisure time activities, based on the context and patterns of user behavior, to recommend relevant places for carrying activities. As another example, [235] presents an activity-aware recommender system for teams of medical professionals working in hospital operating rooms. It suggests relevant virtual actions (e.g., retrieval of information resources and initiation of communications with professionals outside the operating rooms) based on the current state of the operation (detected from sensor data) and considering similar past situations, by using ML techniques.

The EventAware [239] system is able to personalize the agenda of users participating in a congress (e.g., recommending them sessions and exhibitors of interest). For the recommendation process, the authors used a tag-based approach which determines the similarity between the tags that describe the items and the areas of interest of the users. To minimize the number of user interactions with the system, it implicitly incorporates some user’s contextual information (e.g., the location and time) and tags obtained from Wikipedia (e.g., tags that describe the items) and LinkedIn (e.g., basic information of the user’s account and his/her areas of interest, represented by tags). In addition, the user provides information about the days that he/she will be at the event. EventAware first applies the pre-filtering paradigm to ignore items that do not match the user’s current context. Then, the tag-based approach is applied on the candidate items to generate recommendations that match the user’s preferences. Depending on the user’s current context, the final recommendation list will be dynamically modified. In [281], a similar approach was previously presented for the music domain in a web-based environment.

In the domain of context-aware music recommendation, the InCarMusic system [254] is a context-aware mobile recommender system that offers music recommendations to the passengers of a car by using collaborative filtering and matrix factorization. As another example related to music recommendation, a novel probabilistic model for the recommendation of songs for daily activities (e.g., studying, running, walking, sleeping, working, and shopping) was proposed in [242]; in this proposal, contextual information collected by mobile devices is
integrated with music content analysis techniques.

A different and interesting example is the intelligent transport system presented in [249], which tackles logistics problems. It is a clustering-based recommender system that suggests the optimal transporter (e.g., cars, trucks, etc.) to deliver a package for the customer in a smart city. During the recommendation process, the model uses contextual features of transporters (e.g., availability, current location, max speed, fuel consumption, year, millage counter, and CO$_2$ emissions), the road congestion, and the customer’s current location.

As a final example, a multimedia recommendation framework, called RecAm, was proposed in [243]. It incorporates contextual information (e.g., the time, health conditions, emotions, calendar and location data, etc.) into the recommendation process. The purpose of this framework is to facilitate the recommendation of multimedia content by identifying the user’s context through adaptive user interfaces in Ambient Intelligent (AmI) environments. For example, a prototype that uses the proposed framework would be able to detect the stress level of a person (by capturing the heart signal) and recommend suitable songs that can decrease the stress level. Besides, the prototype could modify the environment of the room (e.g., adjusting the volume of the music and the light level) according to the preferences and current context of the user, by using AmI interfaces.

In Figure 13, we show an overview of different types of items that have been considered in mobile CARS. In addition, we present the percentages of works that exploit these items (considering an overall of 29 representative papers). The figure clearly shows that CARS for mobile environments have mainly focused on three types of items: POIs, restaurants, and movies.

5. Conclusions and Open Issues

In this paper, we have provided an extensive survey of AI-based mobile CARS from an information management perspective. We have introduced the technological context needed to facilitate the understanding of context-aware
recommendations in mobile environments, including mobile computing, the main features of traditional recommendation systems, and context-aware recommendation systems. In addition, we have examined in detail pull and push approaches of mobile CARS that exploit artificial intelligence techniques. We have also described some relevant examples of mobile CARS for different application domains.

In this study, we have identified several promising artificial intelligence models that have been applied to the development of traditional RS. Subsymbolic artificial intelligence approaches (where the models are not explicitly represented through elements such as formulas or rules, but learned from experience) are prevalent nowadays [282], and in particular deep learning approaches [283, 284, 285] have been recently explored in the specific context of mobile CARS (e.g., Auto-Encoding [232], Feed-Forward Neuronal Networks [251].
Convolutional Neural Networks [255], and Recurrent Neural Networks [252]).
We could expect an increasing consideration of these types of techniques, as well as improvements of more classical ones, to offer more relevant customized recommendations to mobile users in a variety of potential use cases. However, the use of these complex neural networks leads to important challenges in the domain of RS [280], such as the large amount of data required for each user during the training phase to achieve high-quality predictions and the difficulty to provide explanations of the items recommended to the users. Hence, traditional classification approaches (e.g., association rules, decision trees, Random Forest, Naïve Bayes, Bayesian Networks, and Conditional Random Fields) are still widely used to estimate user preferences and to recognize the activity being performed by the user [224, 233, 234, 235, 240, 242, 247, 248, 250, 281]. Moreover, Canopy and k-means clustering techniques have been among the most frequently used approaches to find groups of similar users based on their profile information [217, 236, 237, 248, 249]. Another observation in this study is that statistical techniques, such as Principal Component Analysis (PCA), Probabilistic Latent Semantic Analysis (PLSA), and Expectation-Maximization (EM), are also widely used [232, 249, 243]. For example, PCA is a relevant statistical technique used to alleviate high dimensionality and sparsity problems in CARS. An overview of these AI models and trends is shown on the left side of Figure 14.

Although the number of proposals involving pull-based recommendation approaches found during the literature review was significantly higher than those considering push-based recommendation techniques, it is expected that push-based solutions will become a more relevant trend in the near future, thanks to advances in mobile computing and its widespread use. Currently, many proactive recommenders use reasoning approaches (e.g., Fuzzy Logic and rules) to handle the uncertainty during the initial assessment phase that determines if the current situation is appropriate to push recommendations (e.g., 234, 236).

From a technological point of view (see the right side of Figure 14), we have noticed that most of the reviewed works have proposed mobile context-aware recommendation applications considering centralized (client-server) archi-
In this scenario, the server hosts the contextual recommendation engine and the needed databases to store information about users, contexts, items, and ratings. Java frameworks such as Mahout and Weka are commonly applied to develop the context-aware recommendation logic (e.g., see [224, 281, 250, 236]). Tools such as MATLAB (with its Fuzzy toolbox) have been successfully used to develop fuzzy inference systems (e.g., [246]). We believe that, in a near future, Python frameworks (e.g., TensorFlow, Theano, Keras, and PyTorch) will be widely used, due to the recent interest to apply deep learning techniques for mobile CARS. We also expect more solutions where at least part of the recommendation engine could be executed in the mobile clients; besides, several mobile devices could share data and tasks as part of a distributed recommendation process.
Currently, in most approaches, the mobile device of a user manages information about the user’s profile and preferences and provides the graphical user interface of the recommendation system, but it does not usually play a relevant role in the recommendation algorithms. The majority of the mobile client applications identified in this review are Android prototypes implemented in Java with Android Studio. CoreLocation, Geo2tag, Geolocation, Google Maps, Google Places, and MapKit, are examples of tools and services used by researchers of mobile CARS to acquire the locations of users and items [26, 240, 245, 246, 274].

As we have seen in this survey, relevant work related to mobile CARS has already been performed. However, there are still significant open issues. Some challenges of context-aware recommender systems are inherited from traditional recommender systems (e.g., data sparsity, cold start, and high dimensionality problems, as well as security, privacy, and spam vulnerability concerns), while others arise as new ones. Among the additional challenges, the following can be highlighted for the general case of CARS [32, 67, 286, 287]:

- The variety of application scenarios and user needs makes it difficult to determine what types of contexts are actually needed in CARS. Hence, the efficient discovery of valid (or suitable) context types for a specific domain is a serious challenge that CARS should overcome, to reduce the difficulty of context acquisition and the computational cost of recommendation algorithms, thus improving the performance accuracy of CARS. According to [30], this challenge can be treated as a problem of feature selection to reduce the dimensionality of the context, and thus make context comparisons more efficient.

- Context acquisition and automatic discovery of dynamic user preferences from several external data sources (e.g., social networks, sensors, RFID data, etc.) is a major research challenge for CARS. The resulting recommendations could be more effective if the characteristics of the dynamic environment were effectively exploited. For this purpose, the use of text mining techniques (applied on users’ reviews, items’ descriptions,
and other user’s texts like posts in social networks), such as sentiment
analysis [288, 289], could play a key role in the design of recommender sys-
tems [290]. As an example, the SenticNet 6 approach detects the polarity
of a text through an ensemble of symbolic and subsymbolic AI tools [291];
this is motivated by the fact that, according to the authors, “Coupling
symbolic and subsymbolic AI is key for stepping forward in the path from
NLP to natural language understanding” [292]. Capturing and considering
the current emotions of a user, to design affective recommenders (emotion-
based recommendations) [293, 294, 295, 296, 297], is also a research avenue
that would benefit from further research. Indeed, as mentioned in [298],
affective computing and sentiment analysis can enhance the capabilities
of recommendation systems.

- Another critical issue for CARS is the development of generic contextual
models. The problem of the current proposals (e.g., [299, 300, 301]) is
that they model information for a very specific application domain (e.g.,
tourism, movies, etc.) or more abstract domains but for specific types of
items (e.g., products, web services, e-learning, etc.), and so their domain-
specific models cannot be easily reused in other recommendation scenarios.
Some proposals try to solve this challenge. As an example, [302] presents
a generic model using an ontology, which can be used in different types of
recommender systems, and models data, context, and the recommendation
process itself. Moreover, a study to try to determine whether a more
generic modeling approach could be applied for CARS was carried out
in [303]: as a result of the study, the authors proposed a novel generic con-
textual model for CARS, which was theoretically evaluated with positive
results.

- Very few proposals in the CARS literature combine the context’s history
and the user’s behavior [304, 305]. Hence, understanding the user’s behav-
iors based on the context’s history could be improved, which may lead to
improve the accuracy of recommendations.
According to [306], the link between cognitive models and recommendation systems has not received enough attention so far. However, decision-making is based on cognitive information [307]. Critique-based recommenders [307, 308, 309, 310] establish a feedback-based recommendation process to refine the initial recommendations provided; in this way, the user can provide critiques of the previous recommendations until a suitable recommendation is received. Moreover, it has been reported that the use of cognitive approaches and linguistic formalism can lead to intelligent information sharing [311, 312]. Although cognitive approaches can be applied for recommender systems in general [313], we believe that they could be particularly relevant to CARS, as the context can play a key role in decision-making.

Besides, we have identified several relevant open research issues for the specific case of mobile CARS, where the existence of a highly-dynamic context and the mobility of users play a key role:

- There is still a gap between CARS and mobile computing. For example, the recommended items are usually considered to be static (e.g., not moving). Besides, some data management techniques for mobile computing, for example approaches for distributed data processing and mobile P2P data management, could be needed for the effective deployment of mobile CARS in certain environments [314]. For example, mixed RS and mobile computing solutions would be useful in the case of a user who is walking down the street and uses a mobile application that suggests to him/her an appropriate taxi in real-time (in this case, both the user and the target items may be moving). Most of the proposed mobile CARS operate on centralized infrastructures (client–server architectures) [231, 235, 236, 237, 240, 241, 242, 243, 245, 246, 248, 249, 274], and so they are subject to the traditional limitations of centralized approaches. Besides, new attributes relevant to mobile users in the context of the COVID-19 pandemic, such as the social distance between people,
should be considered.

- **User interfaces designed for recommendation purposes** (explicit or implicit recommendations) should be simple and easy to understand. However, very few studies have evaluated the usability of interfaces for recommender systems or have studied in depth the best way to present the information to recommend. Usability aspects could be particularly relevant in the case of recommender systems designed for mobile users, as the users may need to interact with a recommender system using a mobile device while on the move. For example, more research could be devoted to the design of critique-based recommenders for mobile users, such as MobyRek and Shopr.

- There is a need for the development of **generic and flexible architectures** that facilitate the creation of context-aware recommender systems for mobile environments. This aspect should be analyzed, given the interest of having a generic solution that can be extended and adapted to different applications and domains. As described in Section 4.1, some context-aware mobile recommender systems have already been developed, but they focus mostly on specific domains (e.g., restaurants, museums, gas stations, supermarkets, foods, etc.).

- Research on **push-based recommender systems** is still in its infancy. Most context-aware recommender systems require users to explicitly express their interests and information needs as a query (explicit request). Currently, due to limitations of mobile devices (e.g., typing data using a small device is inconvenient), a key challenge is how to **proactively deliver relevant recommendations** to the user’s mobile device.

- There is **no common methodology established for the evaluation of mobile CARS**. Some works deploy real implementations and test the usability of the systems by asking the users to fill out questionnaires to capture the user satisfaction. The lack of
public datasets available and the emergence of new evaluation measures different from the classical ones (e.g., MAE, RMSE, precision, recall, and F-measure score), such as the combination of metrics (e.g., combining the accuracy and the diversity with the latency) or the incorporation of context parameters in existing measures, are currently critical challenges [322]. As claimed in [323], determining what makes a good recommendation is not an easy task and the expected utility of a recommendation is a function of the item’s features, the context, and the user’s goals (indeed, “What we like may not be what we choose”). Besides, many of the mobile CARS presented in this paper are evaluated with custom-built datasets (e.g., [232, 241, 244, 248]) and through survey scores (questions) [26, 236, 237, 239, 241, 242, 243, 245, 246, 250, 273, 274], which limits the generality of the results.

- There are not many practical mobile context-aware recommendation applications, especially in real business domains. Indeed, most research on CARS has been conceptual and there has been little work done on developing practical applications for CARS [32]. Generally, researchers implement a context-aware recommendation model, which is tested (using datasets) and compared with other models proposed in the literature of the same domain.

- Another current challenge is to guarantee a suitable privacy protection. Mobile CARS must be able to include privacy protection techniques that protect users’ personal and contextual information (e.g., location, preferences, etc.).

- Only a few approaches have recently considered that mobile users may move in groups. As this situation may happen frequently, it is relevant to study in more depth how to develop mobile CARS that offer item recommendations to groups of users (e.g., a tourism group) [243, 324]. These approaches should consider the preferences of the different persons
in the group and criteria such as fairness for all the users and maximum utility for the group.

In Figure [15] we show a summary of relevant challenges hierarchically related considering traditional RS, CARS and mobile CARS.

<table>
<thead>
<tr>
<th>Traditional RS</th>
</tr>
</thead>
<tbody>
<tr>
<td>- The cold start problem.</td>
</tr>
<tr>
<td>- The sparsity problem.</td>
</tr>
<tr>
<td>- The scalability of recommendation algorithms.</td>
</tr>
<tr>
<td>- The use of distributed architectures.</td>
</tr>
<tr>
<td>- Recommending to user groups.</td>
</tr>
<tr>
<td>- The lack of contextual information may lead to unsuitable recommendations.</td>
</tr>
<tr>
<td>- The support of multi-criteria ratings.</td>
</tr>
<tr>
<td>- Privacy-protection of users.</td>
</tr>
<tr>
<td>- The design of interfaces that operate on mobile devices.</td>
</tr>
<tr>
<td>- The proactive recommendation of items.</td>
</tr>
<tr>
<td>- The diversity of recommended items.</td>
</tr>
<tr>
<td>- The serendipity.</td>
</tr>
<tr>
<td>- The delivery of explanations of the reasons for the recommendations provided.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CARS</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Efficient discovery of suitable context types.</td>
</tr>
<tr>
<td>- Context acquisition and automatic discovery of dynamic user preferences.</td>
</tr>
<tr>
<td>- Development of generic contextual models.</td>
</tr>
<tr>
<td>- Understanding the user's behaviors based on the context history.</td>
</tr>
<tr>
<td>- The gap between CARS and cognitive models.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mobile CARS</th>
</tr>
</thead>
<tbody>
<tr>
<td>- The gap between CARS and mobile computing.</td>
</tr>
<tr>
<td>- User interfaces designed for pull and push recommendations.</td>
</tr>
<tr>
<td>- Generic and flexible architectures.</td>
</tr>
<tr>
<td>- Push-based recommender systems are still in their infancy.</td>
</tr>
<tr>
<td>- There is no common methodology established for the evaluation of mobile CARS.</td>
</tr>
<tr>
<td>- There are not many practical mobile context-aware recommendation applications.</td>
</tr>
<tr>
<td>- Recommending to mobile user groups.</td>
</tr>
</tbody>
</table>

Figure 15: Summary of challenges related with RS, CARS and mobile CARS.
Acknowledgments

This research has been supported by the project TIN2016-78011-C4-3-R (AEI/FEDER, UE) and the Government of Aragon (Group Reference T64.20R, COSMOS research group). We are also exploring mobile CARS in the context of the TRAFAIR project (2017-EU-IA-0167), co-financed by the Connecting Europe Facility of the European Union.
References


   URL http://www.ebay.com


[16] B. Smith, G. Linden, Two decades of recommender systems at Amazon.com, IEEE Internet Computing 21 (3) (2017) 12–18. doi:10.1109/MIC.2017.72


[57] L. Kazemi, C. Shahabi, Geo Crowd: Enabling query answering with spatial crowdsourcing, in: 20th SIGSPATIAL International Conference on


[100] F. Clarizia, S. Lemma, M. Lombardi, F. Pascale, A mobile context-aware information system to support tourism events, in: M. Au, A. Castiglione, K.-K. Choo, F. Palmieri, K.-C. Li (Eds.), 12th International Conference on Green, Pervasive, and Cloud Computing (GPC), Vol. 10232 of Lecture


IEEE Communications Surveys & Tutorials 18 (1) (2016) 68–93. doi:10.1109/COMST.2014.2381246


[131] J. Xu, Y. Yao, H. Tong, X. Tao, J. Lu, RaPare: A generic strategy for cold-start rating prediction problem, IEEE Transactions on Knowledge and Data Engineering 29 (6) (2017) 1296–1309. doi:10.1109/TKDE.2016.2615039


doi:10.1007/s10207-007-0049-3

doi:10.1109/EEE.2004.1287319

doi:10.1145/1141277.1141515


doi:10.1109/TCE.2006.1605022

doi:10.1016/j.eswa.2005.11.010

doi:10.1109/UBICOMM.2008.16


[181] P. Pirasteh, D. Hwang, J. J. Jung, Exploiting matrix factorization to


84


[195] T. D. Pessemier, J. Dhondt, L. Martens, Hybrid group recommendations


[202] X. Chen, Y. Zhang, Z. Qin, Dynamic explainable recommendation based on Neural attentive models, in: 33th AAAI Conference on Artificial In-


W. Woerndl, J. Schlichter, Introducing context into recommender systems,


88


90


[238] Y. Zheng, X. Xu, L. Qi, Deep CNN-assisted personalized recommendation


[258] D. Lian, C. Zhao, X. Xie, G. Sun, E. Chen, Y. Rui, GeoMF: Joint geographical modeling and matrix factorization for Point-Of-Interest recommendation, in: 20th ACM SIGKDD International Conference on Knowl-


X. Li, D. Han, J. He, L. Liao, M. Wang, Next and next new POI recom-


[278] W. Zhang, J. Wang, Location and time aware social collaborative retrieval for new successive Point-Of-Interest recommendation, in: 24th ACM Con-


[301] G. Sielis, C. Mettouris, G. Papadopoulos, A. Tzanavari, R. Dols, Q. Siebers, A context aware recommender system for creativity support


be held in Chiang Mai (Thailand) but finally reconvered to an online conference due to the COVID-19 pandemic. doi:10.1145/3428690.3429164


