A RECOMMENDATION MODEL FOR ENERGY-EFFICIENT DATA MINING SERVICES BASED ON DATA PROPERTIES

by

Zainab Al-Zanbouri

M.Sc. in Computer Science, Ryerson University, Toronto, Canada 2015

M.Sc. Computer Science, Liverpool John Moores University, United Kingdom, 2012

B.Sc. in Computer Science, University of Technology, Baghdad, Iraq, 1991

A dissertation

presented to Ryerson University

in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

in the program of

Computer Science

Toronto, Ontario, Canada, 2019

AUTHOR'S DECLARATION FOR ELECTRONIC SUBMISSION OF A DISSERTATION

I hereby declare that I am the sole author of this dissertation. This is a true copy of the dissertation, including any required final revisions, as accepted by my examiners.

I authorize Ryerson University to lend this dissertation to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this dissertation by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my dissertation may be made electronically available to the public.

Abstract

A RECOMMENDATION MODEL FOR ENERGY-EFFICIENT DATA MINING SERVICES BASED ON DATA PROPERTIES

Doctor of Philosophy, 2019

Zainab Al-Zanbouri

Computer Science

Ryerson University

Currently, there is a big increase in the usage of data analytics applications and services because of the growth in the data produced from different sources. The QoS properties such as response time and latency of these services are important factors to decide which services to select. As a result of IT expansion, energy consumption has become a big issue. Therefore, establishing a QoS-based web service recommender system that considers energy consumption as one of the essential QoS properties represents a significant step towards selecting the energy-efficient web services.

This dissertation presents an experimental study on energy consumption levels and latency behavior collected from a set of data mining web services running on different datasets. Our study shows that there is a strong relation between the dataset properties and the QoS properties. Based on the findings from this study, a recommender system is built which considers three dimensions (user, service, dataset). The energy consumption values of candidate services

invoked by specific users can be predicted for a given dataset. Afterwards, these services can be ranked according to their predicted energy values and presented to users.

We propose three approaches to build our recommender system and we treat it as a context-aware recommendation problem. The dataset is considered as contextual information and we use a context-aware matrix factorization model to predict energy values. In the first approach, we adopt the pre-filtering model where the contextual information serves as a query for filtering relevant rating data. In the second approach, we propose a new method for the pre-filtering implementation. Finally, in the last approach, we adopt the contextual modeling method and we explore different ways of representing dataset information as contextual factors to investigate their impacts on the recommendation accuracy.

We compare the proposed approaches with the baseline approaches and the results show the effectiveness of the proposed ones. Also, we compare the performance of the three approaches to discover the best-fit approach when being measured using different metrics. Both prediction and recommendation accuracy of the proposed approaches are significantly better than the baseline models.

Acknowledgements

I would like to express my deep appreciation to my supervisor, Dr. Chen Ding for accepting me as a Ph.D. student and for her patient guidance, valuable discussions, continuous encouragement, and for her useful critiques throughout my research and thesis writing.

I would like to thank my thesis examination committee, Dr. Patrick Hung, Dr. Olivia Das, Dr. Abdolreza Abhari and Dr. Ali Miri for taking the time and making the effort for reviewing this thesis.

My grateful thanks are also extended to all my colleagues, academic and technical staff in the Computer Science department at Ryerson University for their support and encouragement during my study.

Further, I would like to express my appreciation to all my family members and friends for their support and encouragement.

Finally, I would like to thank my two daughters *Yusur* and *Jana* for their understanding and inspiration throughout my study.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

CAMF	Context-Aware Matrix Factorization
CARS	Context-Aware Recommender Systems
CF	Contextual Filtering
CM	Contextual Modeling
CO_2	Carbon Dioxide
DEV	Deviation
h	Hours
ICS	Independent Context Similarity
ICT	Information And Communications Technology
IT	Information Technology
LCS	Latent Context Similarity
LR	Linear Regression
MCDM	Multi-Criteria Decision Making
MCS	Multidimensional Context Similarity
MF	Matrix Factorization
MLP	Multi-Layer Perceptron
QoS	Quality Of Service
RMSE	Root Mean Squared Error
RS	Recommender System
RT	Reptree
SGD	Stochastic Gradient Descent
SIM	Similarity
TF	Tensor Factorization
VPN	Virtual Private Network Technology
W	Watts
Wh	Watt-Hours

Chapter 1
Introduction

1.1 Background and Problem Statement

The expansion in the Information Technology (IT) has significantly raised the energy consumption levels in the last century. Therefore, there is a growing trend that more researchers have started to focus on the amount of energy consumption associated with the IT field and the possible solutions for reducing these levels. More specifically, reducing the energy consumed by software systems via designing and selecting green software that consumes less energy represents a vital step towards the Green IT goal. Many research works have been done in this area, seeking for more energy-efficient IT. The main concern behind this trend is the noticeable increase in the amount of energy consumption resulting from the IT industry and consequently the increased energy bills. It is expected that the data center energy consumption in the US only will rise from 91 TWh[1] in 2013 to 139 TWh in 2020, with the energy bill increasing from $9.0 billion USD to $13.7 billion USD [1].

In the past decade, the number of web services publicly available is increasing continuously on the internet. More and more companies have migrated or are considering to migrate their legacy software systems into the cloud-based service systems [2]. It could be hard for a user to choose the proper web service among a large number of available candidate services. To address this issue and help the users choose services best fit for their requirements and preferences, web

[1] TWh: Terawatt Hour(s) is an electrical energy consumption rate equivalent to a trillion watts consumed in one hour.

service recommender systems have been successfully built and have achieved good results based on user's preferences and experiences on QoS (Quality of Services) [3].

QoS values are important criteria in the web service selection process [4]. Most of the existing QoS-based web service selection approaches did not take into consideration the energy consumption levels. Therefore, there is a need to build a service recommender system that considers the amount of energy consumed by web services, and accordingly select and recommend energy-efficient services.

Energy-efficient web service refers to a service that uses less energy to provide the same function as other services. Energy-efficient services have many potential benefits such as reducing CO_2 emissions (which has a negative impact on the environment) and decreasing the total energy cost per service. Reducing the energy consumption of web services can be achieved by predicting the energy consumption values of the candidate services that satisfy the user requirements. Once the energy prediction has been made, energy-efficient web services can be selected and recommended.

Generally, the energy consumption associated with data mining web service invocations, as well as other QoS values such as latency and response time, can be largely affected by the input dataset [5]. For example, if we have two datasets varying on different properties (e.g., the number of attributes, number of instances), the associated QoS values (e.g., energy consumption) of the data mining services that are used to process these datasets will be different as well [5]. Therefore, it is crucial to take the dataset information into consideration when recommending such services. It can assist in identifying and promoting the use of those more energy-efficient web services for a given input dataset. In the related research area, there is a limitation of not

considering dataset information in the recommendation process; hence our presented work in this dissertation represents a vital contribution to this research area.

There are various types of data mining services; in this work, we are focusing on clustering services. Data clustering refers to an unsupervised learning technique; it is the process of grouping data objects in a way that the objects in one group are similar to each other while different with the ones in other groups [6].

Context represents any information that can be used in characterizing the situation of an entity. In this dissertation, we use the following terminology for context related terms. The term *contextual dimension* is used to denote the contextual variable, e.g., "number of attributes" of the dataset is one contextual dimension in our work. The term *contextual condition* refers to a specific value in a contextual dimension, e.g., "Small" and "Large" are two contextual conditions for the contextual dimension "number of attributes". Here, "Small" defines the range of values for "number of attributes". For example, if the maximum number of attributes of all datasets is 900, and the minimum number of attributes of all datasets is 1, we may divide the (min, max) interval to three ranges – (1, 300) as "Small", (301, 600) as "Medium", (601, 900) as "Large". A *contextual situation* is a set of contextual conditions, e.g., {Small, Small, Numeric} is the contextual situation corresponding to three contextual dimensions namely, "number of attributes", "number of instances" and "data type".

In this dissertation, we may also use the term *Contextual factor* alternately with the term *contextual dimension* to refer to any information that is related to the user or item and can be integrated into the recommender system, for example, location, time and companion (such as

alone or *with friends*). Figure 1.1 illustrates the conceptual model representing relation between different terms in the *context* domain.

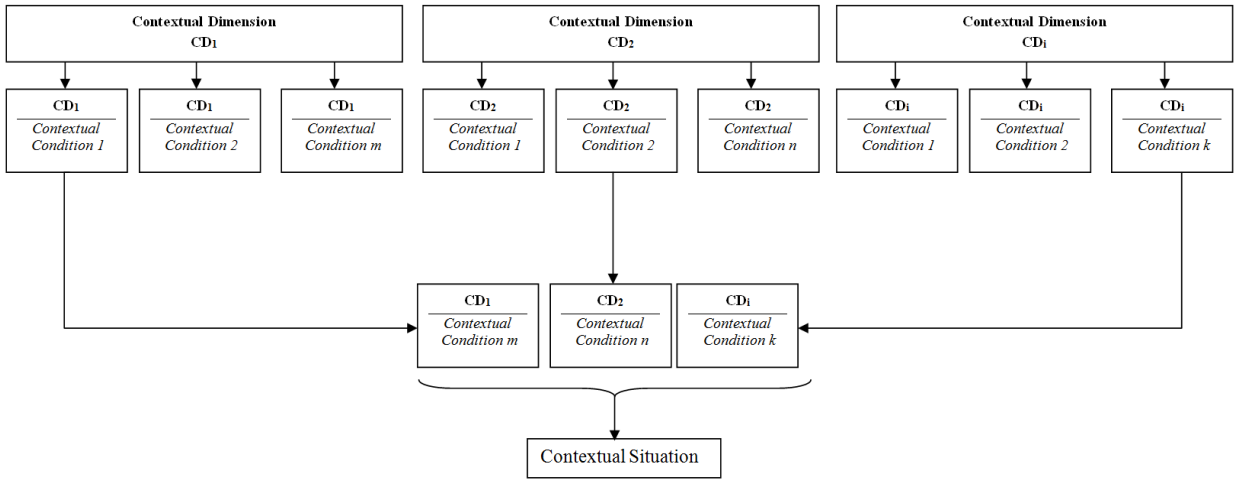

Figure 1.1: The conceptual model representing relation between context-related terms.

As we can see from Figure 1.1, each contextual dimension *CD* can include various numbers of contextual conditions. For example, we have *i* contextual dimensions, and *m*, *n* and *k* refer to the number of contextual conditions in each of the contextual dimensions CD_1, CD_2 and CD_i respectively.

In the traditional recommender system, there is a limitation of not considering the context in which an item is consumed. To address this limitation, Context-Aware Recommender System (CARS) has been proposed for incorporating additional relevant information (context) into the recommendation engine. Using context has shown to be able to provide more accurate predictions and more relevant recommendations [7-10]. Some of the web service recommender systems consider contextual information such as location, time, etc [11].

Our work presented in this dissertation focuses on data mining services running as web services, where multiple users from different locations are invoking various services and using different datasets. We propose to include dataset information as contextual factors that can be integrated into the existing service recommender system to assist in recommending energy-efficient web services. Although there are many previous works in the literature that focused on incorporating context in recommender systems, to the best of our knowledge, our work represents the first attempt of incorporating data as contextual information which enhances both prediction and recommendation accuracy.

1.2 Dissertation Statement

This dissertation includes the construction of a service recommender system which aims to recommend the service with the least energy consumption from a list of services having the same functionality (clustering). The recommendation is constructed based on the properties of the used dataset. Various approaches are used and compared towards this goal, and a comparison is made with the existing baseline models in order to evaluate the best proposed approach for our research problem.

1.3 Objectives

In this section, we introduce the main objectives we would like to achieve in this work.

- Examine the factors that might affect energy consumption values. In this dissertation, we need to identify the factors that might have impacts on web services' energy consumption. We need to study these contextual factors, identify the influential ones, and add them into our context-aware recommender system. Since we are looking for energy-efficient services, knowing these factors can assist us in recommending the optimal service to the target user. In the current

work, we choose to focus on the properties of the used datasets, although there are other factors that may also impact the energy values.

- Develop a context-aware recommender system for recommending energy-efficient web services based on the findings from the previous objective. Most of the existing web service recommender systems are not considering data in the recommendation process; therefore, one of our objectives in this dissertation is to build a context-aware service recommender system that takes into account dataset properties as one of the contexts in recommending the best services.

- Evaluate the prediction and recommendation accuracy of our proposed recommender system. In this work, we propose three different recommendation approaches. We need to evaluate these proposed approaches to figure out the improvements in predicting and recommending energy-efficient web services over the baseline models, and also compare the three approaches to identify the best-performing one.

1.4 Proposed Approaches

In this dissertation, we propose a context-aware recommender system for energy-efficient data mining services (clustering services). In our solution, we introduce three different approaches namely: similarity-based pre-filtering (SBPF), weight-based pre-filtering (WBPF) and dataset-based contextual modeling (DBCM). In the first two approaches, we adopt the contextual filtering (pre-filtering) model based on the assumption that if the contextual information is similar then the recommendation has to be similar as well [12]. We did that via using the context-aware matrix factorization model and applying the pre-filtering technique. The SBPF approach is presented in our paper [7].

More specifically, in the second approach (WBPF) we introduce a new method of applying the pre-filtering technique, where we consider the context similarity as a weighted factor that can be used to adjust the rating values before applying the pre-filtering technique. In the last approach (DBCM), we adopt the contextual modeling approach, where the contextual information represented by the dataset properties can be incorporated directly into the recommendation process. This approach is presented in our paper [13].

From the evaluations we obtained from our experimental results, we found that all the three proposed approaches outperform the baseline model. Moreover, we found that the weight-based pre-filtering approach is the one that best fits our problem for recommending energy-efficient service based on the used datasets.

1.5 Assumptions and Scope of the Work

In this work, we assume that the measurement of energy consumption associated with web service invocation can be done by the service hosting provider [14]. Also, we assume that the service providers have the motivation to perform this measurement because it has a clear benefit in reducing the amount of energy consumption and thus decreasing the bills accompanying the consumed energy for each single web service invocation.

In terms of the scope of this work, currently, we are focusing on only one type of data mining services, namely clustering services, although the proposed solution potentially can be applied to any type of data mining services. Furthermore, regarding the QoS properties under study, we focus on energy consumption in our proposed recommender system, because our main target is to recommend energy-efficient services. However, other QoS values can be easily added to our proposed solution such as latency, response time and reliability.

The dataset properties that we are focusing on in this dissertation are number of instances and number of attributes of the dataset. In addition, we consider data type of the dataset as a third property. In relating to the data type property of the used dataset, we focus on two data types, namely numeric and nominal. Although we did not consider having images or photos as a data type in this work, our introduced recommender system can be extended to include any data type. Regarding the contextual factors that we have integrated into the recommender system, we have only considered the dataset properties, with different combinations of them. We did not consider other contextual factors such as the invocation time of the web service.

In this work, we assume that there is only one invocation at a time, and we didn't consider the case of parallel invocations (e.g., request on the same service from multiple users at the same time). Because of this assumption, the design of our measurement framework can be simplified, and we measure the amount of energy consumption associated with each single web service invocation. Furthermore, we did not consider the distributed implementation of data mining algorithms or the case of distributed dataset.

1.6 Motivating Examples

Recently, there are many research efforts that focus on reducing the amount of energy consumption associated with the IT field. This type of research has many valuable benefits especially in relating to cutting down the energy bills for big IT companies. In this section, we give two examples to illustrate the importance of considering the energy consumption as the main property for recommending the optimal energy-efficient services to the target users based on the properties of the used datasets.

In the first example, assume that a company wants to find a clustering service to do user segmentation on their customer data. Since they do not want to save the customer data in the public cloud due to the security concern, they plan to run the service on their own private cloud. In order to select such a clustering service best for their dataset, they rely on a recommender system to recommend a service based on the historical information of the service performance in the past. This recommender system provides the recommendation by finding the similarity between the target dataset and the historical datasets and then provides the best service among many available services. The optimal service can be selected depending on different QoS values such as response time, latency or the energy consumption associated with the service invocation. In this case, the company is going to run the service on their own private cloud, energy consumption is a big concern, and therefore our recommender system could be a perfect match for them because the energy-efficient services can be recommended.

In the second example, assume that an e-health application tracks a user's daily activities. The application is run in an edge computing environment, in which the main processing is done on the central cloud server, and lower-level processing and analysis are done on the edge servers. Since there is a lot of data collected every day, to reduce the network transmission and the workload of the central server, data from one user will be clustered on the edge devices first, then further processing will be done on the edge server and eventually on the central server. In this case, for clustering services running on edge devices, energy-efficiency is crucial and reducing its level is important as it will assist in decreasing the overall energy consumption levels and consequently reduce the corresponding energy cost. Therefore, applying our recommender system can help recommend the clustering service with the least amount of energy consumption. Based on the similarity degree between the users' datasets our system can be used

first to predict the amount of energy consumption required for each available clustering service and then rank the services based on the amount of predicted energy consumption. Finally, the service that exhibits the lowest energy level can be considered as the optimal service which can be recommended to cluster the user's activity data.

In the two examples above, we can notice that recommending a list of services with the least amount of energy consumption would bring benefits to the service provider in these situations because it can help in cutting down the cost of the services and pushing the sustainability process forward.

1.7 Dissertation Organization

The rest of the dissertation is organized as follows:

- In Chapter 2 we discuss the background information in relating to the scope of this dissertation, also we present the related work in various relevant areas.
- In Chapter 3 we give an overview of the proposed recommender system and we introduce our proposed framework for recommending energy-efficient data mining web services.
- Chapter 4 discusses the impacts of different factors on energy consumption via introducing our experimental study on energy consumption.
- Chapter 5 presents the methodology used in each of the proposed recommendation approaches.

- Chapter 6 discusses the experiment design and set of experiments related to each proposed approach. Moreover, in this chapter, we present the evaluations of the proposed approaches against the baseline models and comparison among the proposed approaches.
- Chapter 7 concludes the dissertation with the future directions of research.

Chapter 2

Literature Review

2.1 Background

2.1.1 Context-Aware Recommender System

In general, the standard recommendation problem starts with a two-dimensional matrix of ratings: User × Item → Rating. The important idea behind context-aware recommender systems is that the user's preferences for items might be considered as a function of the context in which those items are encountered. In order to incorporate context, we need to estimate user preferences by using a multidimensional rating function R: User × Item × Context → Rating [15]. Usually, there are two methods to implement this. The first method is to use context as filters in the algorithm, which is known as Contextual Filtering (CF). The second method is to use context as one part of the predictive functions in the recommendation process, and this method is known as Contextual Modeling (CM).

In the literature, there are different methods used to generate context-aware recommendations: pre-filtering, post-filtering and contextual modeling [16]. In pre-filtering, the contextual information is used for selecting the relevant set of data records. Post-filtering, on the other hand, initially ignores the contextual information and ratings are predicted using any traditional two-dimensional recommender system on the entire dataset. Afterwards, the resulting set of recommendations is contextualized for each user by using the contextual information. In contrast, in the case of contextual modeling, the contextual information is used directly in the modeling techniques as part of rating estimation.

2.1.2 Matrix Factorization and Context-Aware Matrix Factorization

Matrix factorization (MF) is one of the most popular algorithms to implement recommender systems. MF aims at factorizing the rating matrix R: $m \times n$ into two $m \times k$ and $n \times k$ dimensional matrices *P* and *Q* respectively, in which *m* represents the number of users, *n* is the number of items, and *k* is the number of latent factors. In this algorithm, both users and items are represented by vectors: $\overrightarrow{p_u} \in P$ and $\overrightarrow{q_i} \in Q$ respectively. These two vectors include values representing the weights on *k* latent factors (hidden factors that contribute to the user's preferences or rating on an item). Usually, the rating matrix *R* includes some missing values because there might be some items that are not used by some users. MF can predict these missing values, and based on predicted ratings give recommendations [17].

Generally, there are two main categories of the models used for implementing context-aware recommender system namely dependent and independent. In relating to the first category, tensor factorization (TF) represents one of the common models in this group. In relating to the latter category, the similarity-based and deviation-based models are the most common models.

In [18], a number of different context-aware matrix factorization techniques were developed to capture the interaction between the ratings and the contextual factors (e.g., weather and place of interest). These techniques measure the relevance of contextual factors on the ratings based on three different assumptions and accordingly three models were developed. These three models represent examples of the deviation-based models.

The first model is called CAMF-C. It assumes that each contextual condition (a specific value in a dimension) has an overall influence on the ratings independently from the item. In other words, the effect of each contextual condition on user ratings is the same for all the items.

Therefore, a single parameter for each contextual condition is introduced in this model and this parameter models how the rating deviates from the basic prediction (without context).

The second model is called CAMF-CI. It assumes that the effect of each contextual condition varies for different items. This model introduces a large number of parameters. For each contextual condition and item pair, a parameter is used to model the deviation of the rating. This model provides better prediction when the context influences differently on the items. The third model is called CAMF-CC. It groups items into categories and assumes that the influence of each contextual condition is the same for each item category. One parameter is used to model the deviation for each contextual condition and item category pair.

In the initial experiments in this work, we tried different models from both dependant and independent categories and we used various deviation-based models where we integrated the dataset properties as contextual factors. Also, we compared the various deviation-based models to discover the model that gives better recommendation results in relating to our introduced research problem.

In [19], a number of correlation-based context-aware matrix factorization methods were developed and claimed to be an improvement over the models proposed in [18]. These approaches measure the correlation between two contextual situations rather than rating deviation. The assumption is that two similar contextual situations for a user will produce similar recommendations for the same user. Three different models were developed, namely independent context similarity (ICS), latent context similarity (LCS) and multidimensional context similarity (MCS). These three models represent examples of the similarity-based models.

The ICS model measures the relationship between two contextual situations as the product of the correlations on different dimensions. It assumes that contextual dimensions are independent and therefore only calculates the correlation between contextual conditions in the same contextual dimension.

On the other hand, in the LCS model, each contextual condition is represented by a vector of weights over a set of latent factors, where the weights are initialized at the beginning and learnt by the optimization process. The dot product between two vectors can be used to denote the correlation between each pair of contextual conditions. The MCS model, on the other hand, represents contextual conditions in a multidimensional coordinate system based on the number of contextual dimensions available in the system. The correlation between two contextual conditions is measured by the distance between two points in the coordinate using Euclidean distance. As reported in [19], MCS outperformed the other two models (ICS and LCS).

In our introduced work, we further examined different types of similarity-based models. We considerd using dataset properties as the main contextual factors that can be added into the existing model aiming to improve the recommendation accuracy. Additionally, in our initial experiments, we compared various similarity-based models in order to discover the model that exhibits the best recommendation result in relating to our research problem.

2.2 Related Work

2.2.1 QoS Prediction and Service Selection

As a result of the huge expansion in the IT field and the adoption of the cloud computing paradigm, the research in the area of web services and their related QoS has become a vital subject. Generally, web services can be differentiated by their functional and non-functional

properties. Multiple services can possess the same function while they may differ in their non-functional properties that are often represented by their QoS values.

QoS values are important criteria for service selection and recommendation. There are many previous works in the literature focusing on using QoS properties in implementing their approaches. A method for predicting the QoS values of data mining services based on the attributes of the input dataset using a meta-learning approach was proposed in [20]. In [21] a method for QoS metrification using Hidden Markov Models was proposed for recommending an optimal path for the execution of user requests. In [22] the author tackled the service selection problem by representing service QoS values as discrete random variables with probability mass functions. A method of service selection called the Correlation-Aware Service Pruning method was proposed in [23]. This method managed QoS correlations by accounting for all services that may be combined into optimal composite services and trimming services which are not the optimal candidate services.

A statistical learning approach for extracting domain-specific QoS features from user-provided service reviews was developed in [24]. The main goal of this approach is to classify user reviews depending on their sentiment orientations as either a positive or negative category. The authors also developed an approach for automatically combine related terms and return the term groups to users, where each term group corresponds to one high-level quality aspect of services. In [25] the importance of QoS properties was presented in different service computing tasks, such as selection, discovery, recommendation, and composition. The authors conducted an evaluation of many real-world services and collected throughput and response time as QoS features. The used web services were tested and the evaluation focused on response time and throughput.

QoS monitoring is considered from a service user's perspective in [26]. The authors described an approach in which monitoring requirements are expressed as queries in a simple language and are processed against continuously arriving QoS data streams. In [27] an approach was proposed for automatically learn and monitor QoS values of web services by using Aspect-Oriented Programming (AOP). Under AOP, a service stub is produced for each evaluated service and an invocation was performed for the evaluation. The proposed approach focused on a subset of common QoS properties that can be measured through invocation, such as availability, response time, execution time, and accuracy. An approach for predicting dynamic QoS values was proposed in [28]. It leveraged both auto-regressive integrated moving average and generalized autoregressive conditional Heteroskedasticity models to capture the volatility of QoS data and predict future values.

An approach for computing the overall QoS values of a composite service depending on the workflow of the service and QoS values of each component service was developed in [29]. Rather than using fixed values, the proposed approach modeled each QoS parameter as a probabilistic distribution to better capture the real-world scenarios. An approach for integrating QoS values and analyzing QoS requirement in reconfigurable web services choreographies was proposed in [30]. It reconfigured a composite service given a QoS goal, with a major focus on latency, throughput, accuracy, and data quality. An architecture model for web service discovery and selection was introduced in [31]. The main component in the proposed work is a machine learning-based methodology for predicting the QoS properties using source code metrics. Both credibility value and previous usage count are used to determine the reputation of the service.

A multi-objective optimization approach was developed in [32] to make trade-off decisions between service's trust value and user's QoS preferences in order to rank candidate cloud services

depending on their match degrees with users' requirements. The authors focused on heterogeneous preference and trust-based service selection. An approach using the semantic web and quality of service model to perform cloud service matchmaking, selection and composition, to fulfill the requirements of an end user was proposed in [33]. The authors developed cloud ontology for providing semantic descriptions to the service provider and requester, so as to automate the cloud service tasks.

There are many differences between our work and the related works presented in this section. First in our work the main focus is on considering the energy-consumption as the main QoS attribute which is used to recommend the optimal energy-efficient service, whereas in the other works their focus is either on accuracy and latency [20] or response time [21]. Second, in our proposed work we used the matrix factorization for prediction while in the other works they based their models on other collaborative filtering approach [20].

2.2.2 Service Recommender System

There are many research works in the literature on service recommender systems. An intelligent recommendation approach named Inverse_CF_Rec was proposed in [34]. For a target user, first a search for his/her opposite users will be implemented and (referred to as ‘‘enemy’’ hereafter); then the possible friends of the target user will be deduced indirectly depending to Social Balance Theory. The optimal services are recommended to the target user based on the derived possible friends of the target user.

An enhanced service recommender system for web services was presented in [35], which takes existing recommendations from services or content providers and based on topics identified by the user's recent browsing history (including past clicks and searches) re-ranks the

recommended URLs for each service, giving an aggregate re-ranked recommendation list. This information can be used by third party services to give more related recommendations and notifications to the user.

In [36] an approach was proposed where pseudo ratings of services are constructed based on user-service interactions and are provided to represent users' preferences on services. Depending on these pseudo ratings, the authors present a web service recommendation approach, which can reduce the cold-start problem by integrating contextual information (e.g., time of updating web services and user location) and an online learning model. In [37] both word embeddings enhanced the hierarchical process and factorization machines are integrated to recommend mobile services to build high-quality Mashup application. The proposed work extended the description documents of Mashup applications and mobile services by using Word2vec tool and derived latent topics from the extended description documents of Mashup and mobile services.

A matrix factorization model was proposed in [38] with deep features learning, that integrated a convolution neural network. The proposed model is named Joint CNN-MF (JCM) and it is capable of using the learned deep latent features of neighbors to derive the features of a user or a service. An approach for solving the service composition problem was proposed in [39] for the case of having two types of requirements of QoS. The authors proposed a composition method and they gave an analysis from composition models of services and from the related quality of experience of web services. After that, they ranked the service candidates and service requests together.

Our work presented in this dissertation is different than the other works in this related area in terms of the type of services we are targeting to recommend to the active user. We focus on data

mining services and the recommendation is based on the properties of the target dataset, whereas in other works, they do not emphasize on data mining services and they do not consider the impact of dataset properties on recommendation results. Many previous works are user-based collaborative filtering system, they recommend by finding user interests through their browsing history [35] or through their derived possible friends [34]. Some of them may focus on improve server dynamic speed [40] instead of recommendation accuracy.

2.2.3 Web Services Energy Consumption

Recently, there are many research and studies that investigate the IT energy consumption and how it contributes to carbon dioxide (CO_2) emissions. The total Information and communications technology (ICT) energy consumption was expected to increase at an alarming rate and the associated carbon footprint would become comparable to that of approximately 2 percent of the total emissions of CO_2 from airline transportation [41].

With the increase of the amount of energy consumption associated with IT systems and its services, energy efficiency is becoming a critical concern in the design, development and management of web service systems. Therefore, there is an obvious need for research that focus on reducing IT energy consumption levels and discovering greener services or applications.

An experimental study on the energy consumption of a few data mining algorithms running on mobile devices was introduced in [42]. Also, the idea of energy-efficient QoS in mobile web applications to balance between good QoS and low energy consumption was proposed in [43]. Further, a decision support procedure to offer a priori, deterministic understanding of power consumption of modular software services in distributed cloud environments was presented in

[44]. Their findings focused on estimating the power consumption of web services' workloads by using an aggregate linear model.

An energy-aware approach for web browsing in 3G based smart phones by reorganizing the computation sequence required for loading web pages so the web browser can initially run the computations that will generate new data transmissions and retrieve these data was proposed in [45]. Afterwards, the authors predicted the user reading time on the web page after it is downloaded. A method for optimizing both web service selection and server dynamic speed scaling by maximizing the quality of service revenue and minimizing energy costs was introduced in [40].

A virtual power meter supported power consumption prediction approach for web service selection that facilitates choosing appropriate services to minimize electrical energy from the overall environment of service-oriented computing applications presented in [46]. The mixed-integer programming technique for optimizing service compositions based on the response time and energy consumption was applied in the proposed approach in [47]. Moreover, there are some studies about measuring the energy consumption of software and applications. An empirical case study of energy consumption by using two MySQL database engines, InnoDB and MyISAM, across 40 releases was conducted in [1]. The results showed that database engines' energy consumption and execution time increase as databases evolve (as a result of the database changes). Also, a number of parameters related to the challenges of designing energy-efficient database clusters were examined in [48].

Furthermore, a tool (EnerQuery) was built on top of a traditional DBMS to capitalize the efforts invested in building energy-aware query optimizers and this tool presented in [49]. A

power-aware query operator placement algorithm that determines which part of a continuous query plan should be executed at the data stream management system and which part should be executed at the mobile device was proposed in [50]. In [51] the authors advocated the transparency of testing in the database field, and then they proposed the use of a repository dedicated for storing testing environment and results.

Energy consumption has been considered as an important QoS property in [43] [46] [47] [52] and [53]. The work in [46] [47] also deals with service selection problem. The major difference between our work and their work is that we focus on data mining services instead of mobile applications or composition services. We have proposed a QoS prediction and service selection algorithm based on our observations on the relationship between QoS values and dataset properties as we presented in [5].

Most of the previous works that have been done in this area are for mobile web services whereas our proposed work is for services running on computers; these two environments are different in terms of the available resources and the performance, e.g., memory size, storage capacity and processing speed. Moreover, in the mobile system in most cases measuring the energy consumption and other system environment variables can be done by using special applications designed for this purpose and they are generally software-based applications.

In contrast, in computer systems, although there are many software solutions available to measure energy consumption, using an external hardware meter (the approach we used in this work) to find the energy consumption associated with a specific task is considered the more accurate and precise solution [5]. The data mining web services under study are implemented as presented in [54] and these services are hosted on a server. Users from various places can send

requests to invoke these services by using different datasets. In the server-side the amount of energy consumption associated with the web service invocation to finish the clustering task is captured and all the related information is stored in one database to be used in the later steps.

The work reported in [42] is most similar to our presented work in the part of recognizing the energy-efficient services as we presented in [5]. However, there are some key differences between the two works. First, in our work (as presented in [5]), we focus on examining the performance of data mining web services in terms of measuring energy consumption, whereas in [42] they examined energy and execution time for data mining algorithms running on mobile devices. Second, we focus on the service selection process and thus we apply a ranking model to rank services based on their predicted QoS values [5] and accordingly we propose our recommender system as we presented in [7] and [13], whereas the work in [42] only predicted energy values of mobile algorithms without ranking. Furthermore, in [42] they studied the impact of using different mobile devices on energy consumption levels, whereas we study the impact of dataset properties and algorithm implementations on energy consumption [5].

2.2.4 Matrix Factorization and Context-Aware Recommender System

In context-aware recommender system, the contextual variables usually represent the attributes of the activity itself while the user or item attributes are considered as content in recommender systems. There are many previous attempts to incorporate different types of information in recommender systems and utilizing them as contexts in the process of recommendation.

An approach for making a context-aware recommendation of places to users based on current weather, time and user's mood was introduced in [55]. A contextual modeling approach

that depends on contextual neighbours was proposed in [56] and presented four different methods to find the contextual neighbours. A context-aware recommendation system for estimating user preferences by sequential predictions was proposed in [57]. Moreover, a context-aware recommendation model named paragraph vector matrix factorization which integrates the unsupervised learning of paragraph embeddings into probabilistic matrix factorization was proposed in [58]. The proposed model can be used to capture the semantic information of the paragraph and can improve the prediction accuracy of the ratings.

In [59] a matrix factorization approach for contextual recommendations was proposed and called the contextual SLIM (CSLIM) recommendation approach. It is based on the sparse linear method which was designed for Top-N recommendations in traditional recommender systems. An algorithm, namely Kernel Context Recommender System, which is a flexible, fast, and accurate kernel mapping framework that recognized the importance of context and incorporated the contextual information using kernel trick while making predictions was introduced in [60].

A deep neural network based recommendation model named Convolutional and Dense-layer Matrix Factorization for a context-aware recommendation, combining multi-source information from item description and tag information was proposed in [61]. The introduced model used a convolution neural network for extracting hidden feature from item description as document and then fused it with tag information via a full connection layer, to generate a comprehensive feature vector. A framework that demonstrated multiple rating prediction algorithms depending on user neighborhood and item neighborhood approaches exploiting explicit and implicit ratings was proposed in [62]. The algorithms incorporated context communities to alleviate the data sparsity problem.

EmoWare (emotion-aware) which is a personalized, emotionally intelligent video recommendation engine was developed in [63] by using a context-aware collaborative filtering approach, where the intensity of users' spontaneous non-verbal emotional response toward the recommended video is captured through interactions and facial expressions analysis for decision-making and video corpus evolution with real-time feedback streams. An intelligent context-aware management framework for cold chain logistics distribution that contains acquisition framework, recommender systems framework, risk management framework, tracing back framework, and user portrait framework of cold chain logistics distribution developed in [64].

A social recommendation method, named SocialConvMF that integrated trust relationship and Convolution Neural Network into Probability Matrix Factorization was proposed in [65]. The proposed method used the trust-aware social recommendation method; then captured contextual information in the document; at the end, it utilized trust-aware and contextual information to do social recommendation.

Matrix Factorization is deemed as an effective technique for recommender systems as it discovers the hidden features behind the relations between users and items. It combines good scalability with predictive accuracy in addition to offering flexibility for modelling various real-life situations [66]. Matrix factorization is simply a mathematical tool that can be used to find out some hidden properties underlying the data. Many of the existing related works deal with a two-dimensional matrix representing users and items. In this case, the standard recommender system is used without considering any contextual information [15].

There are some research works that have been done by using MF and considering contextual information [67][68]. A correlation-based context-aware recommender system using matrix

factorization algorithm was built in [69]. A context-aware recommendation algorithm that extended matrix factorization and modelled the interaction of the contextual factors with item ratings was presented in [18]. A fuzzy mapping relationship between contexts and latent factors and made use of the contextual information to initialize user/item feature vectors was established in [68].

Tensor Factorization model is another tool that has been used in the recommender system. TF is mainly used to present three-way matrices and predict missing entries. We can look at TF as a multi-dimensional form of MF [70]. TF approach is not popular in practical application for its high computational cost. In the existing web service recommender systems, some of them did not consider the contextual information. An approach for performing the prediction of unknown QoS values using fuzzy clustering method with the calculation of the user similarity was proposed in [71]. A privacy-preserving framework through data obfuscation techniques was presented in [72] and developed two representative privacy-preserving QoS prediction approaches in their framework.

There are also many recent service recommender systems that consider the contextual information in the recommendation process. In [73] the authors studied the problem of dynamically modelling the quality of web services and they proposed a hybrid system designing framework for dynamic service quality prediction. In [12] the authors proposed to have context similarity as an alternative contextual modeling method and tested various methods to represent context similarity and integrated it into the recommendation process. A location-aware personalized collaborative filtering method for web service recommendation where both locations of users and web services are influential factors in the user or service selection process was proposed in [11].

Our proposed work is different from the existing work as we focus on a new dimension representing dataset properties as contextual factors to be integrated into the recommendation process in order to enhance the prediction and recommendation accuracy, whereas in the other related work they used different dimensions such as current weather, time and user's mood [55]. Also, in our work we incorporate the contextual information by applying either contextual-filtering or contextual modeling approaches, whereas in the other related work they used fuzzy mapping relationship between contexts and latent factors [68]. Furthermore, in our proposed work we perform the prediction of the unknown QoS values (energy consumption) by using Matrix Factorization model with the calculation of the datasets similarity whereas, in the other previous works they performed that by using fuzzy clustering method with the calculation of the users' similarity [71].

2.2.5 The Gaps in the Related Work

Although there are many previous works in this research area, we found that there are still some significant gaps. From the previous works in the area of service recommender system, we have learned that generally the optimal services are recommended to the target user based on different available information such as the possible friends of the user [34] and the user-service interactions which represent the users' preferences on services.

Many different QoS properties have been considered in the past, such as response time, availability, accuracy and throughput. We found that most of the existing service recommender systems did not take into consideration the energy consumption level as a QoS attribute in the recommendation process. Considering the need for green computing, it is desirable to build a service recommender system that measures the amount of energy consumed by web services and

accordingly select and recommend energy-efficient services. This is one of the key contributions of our proposed work.

We have learned that the context-aware recommender system can be used for recommending an item or service to the target user based on different available contextual information such as time, companion and weather conditions. By integrating the contextual information into the recommendation engine, most of the previous works showed that there is an obvious enhancement in the recommendation accuracy over the standard recommender system (where the context is ignored).

Although there are many previous works in the literature that focused on incorporating different types of information and using them as contexts in the recommender systems, to the best of our knowledge, our work represents the first attempt of incorporating data or more precisely dataset properties as contextual information that helps enhance both prediction and recommendation accuracy for service recommender systems.

Chapter 3

Overview of the Proposed Recommender System

In this chapter, we introduce our proposed framework for recommending energy-efficient data mining web services.

3.1 Recommending Energy-efficient Data Mining Services

3.1.1 Energy Measurement Framework

In the literature, there are different methods available to measure energy consumption in computer systems and applications. Some methods depend on using external hardware devices such as power meter device (Watts UP [74]) while other methods depend on using special software tool that estimates the power consumption of the computer such as the Joule meter [75]. The Joule meter is a tool used to track computer resources, such as CPU utilization and screen brightness, and then estimate the power usage. Generally, using a power meter to capture the amount of energy consumption shows more precise measurements compared with the solutions based on using software tool.

In our work, in order to measure the energy consumed by each single web service invocation, we designed an energy measurement framework for this purpose that utilized a power meter – Watts up? PRO-ES4[74]. This meter has a set of API-communication protocols which can be used in developing applications that communicate directly with the meter; the meter has accuracy +/- 1.5%. The maximum sampling speed is 1 sample per second. The meter can measure and record 18 various parameters such as Current Watts, Cumulative Watt-Hours,

Maximum Watts, etc. In our work, we chose the Cumulative Watt Hours (Wh) parameter as it represents the energy measurement unit and this parameter contains the cumulative measurement starting from the beginning of the logging process until the end of logging.

The introduced energy measurement framework is designed to measure the energy consumed by a data mining service. Furthermore, the energy measurement is mainly built on the concept of measuring the cumulative Watt-Hours which is determined by multiplying the real power a device uses in watts (W) by the amount of time that device is on or running a process in hours (h). The total energy is then given in watt-hours as shown in equation (3.1).

$$\text{Energy} = \text{Power} \times \text{Time} \tag{3.1}$$

The energy measurement in our presented work represents the energy consumed by the invoked web service as well as the network transmission. We consider energy consumption as the main QoS property in this work. Therefore, we used our implemented framework to capture this property for each service invocation.

The major challenge in the energy measurement process is how to determine the exact measurement in a specific piece of program or service. In order to overcome this challenge, we wrote a java program to control the measurement process so it starts and ends with each service invocation. In the real application setting, the energy measurement part can be done on the service provider's servers. Since energy efficiency will largely benefit the service providers, they would most likely want to do it in order to reduce the cost of their web services.

3.1.2 The Proposed Recommendation Framework

In our proposed work, we have used the test-bed system implemented in [54] as it has implementations of the data mining services under study. Also, this test-bed system can be used to collect the QoS values for the invoked web services (which we can add in the future into our presented work). In [54] the authors used three open source data mining packages, namely, WEKA, R and Apache Commons Machine Learning. They developed RESTful web services on three categories of data mining algorithms, including clustering, classification, and association rule mining (in our work we used only clustering services). They used different libraries such as Weka libraries, R Studio, R programming libraries and Apache machine learning libraries [54].

We integrate our proposed energy measurement framework into the test-bed system in order to build a complete system that can be used to 1) invoke a web service, 2) monitor any related standard QoS values and 3) measure the associated energy consumption. In order to collect the QoS values associated with each web service invocation, once the user sends a request to invoke a web service and provides the dataset properties, our system starts the process of monitoring any related QoS values such as latency and response time based on the test-bed system [54]. At the same time, the system captures the amount of energy consumption associated with this invocation from the time of starting the service until the end of the service by using our proposed energy measurement framework. Figure 3.1 illustrates the architecture model of our proposed recommender system.

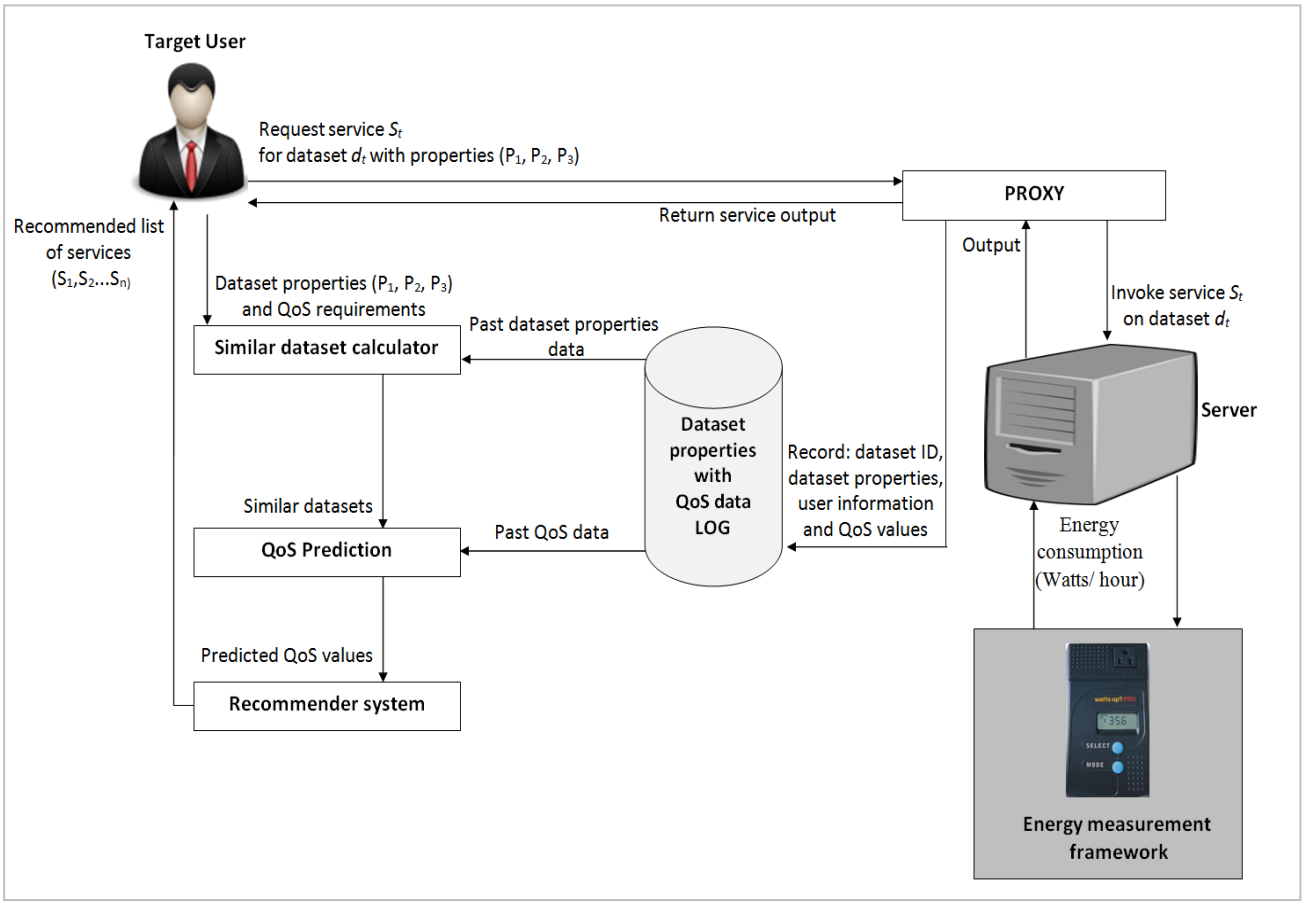

Figure 3.1: Architecture model of our proposed recommender system.

The main components in the proposed framework are listed below:

- **Dataset properties with QoS data log:** this is the central database where the attributes of the datasets as well as the QoS values associated with the services invoked on these datasets are stored. It includes the following parts:

  1- QoS repository: the database that keeps track of the energy consumption values for the invoked services. Each record includes user, invoked web service, dataset ID and associated energy value. In relating to the QoS values, our proposed system can be used to measure and include any associated QoS properties such as latency, response time and

reliability in addition to energy consumption. In this dissertation, since our main goal is to recommend energy-efficient web services, we only consider energy consumption as the representative QoS attribute. However, all the other QoS properties can be included in the recommender system by using utility functions [5] or by using another multi-criteria decision making (MCDM) algorithms [76]. As we stated that in our work presented in [5], where we consider both energy consumption and latency. Energy consumption refers to the amount of energy consumed for a service invocation and the unit of measurement is watt/hour. Latency is the measurement of the time taken by a service to process a given request and the unit of measurement is millisecond (ms).

2- Dataset repository: the database that saves dataset properties under study, including the number of instances, number of attributes and data type. For every service request made, the system records the associated properties of the input dataset alongside the dataset ID.

- **Similar dataset calculator:** the component of calculating the similarity degree between the properties of any two datasets.
- **QoS Prediction:** the process of predicting the missing QoS values (energy consumption). For this purpose, we proposed 3 approaches, namely similarity-based pre-filtering (SBPF), weight-based pre-filtering (WBPF) and dataset-based contextual modelling (DBCM), as illustrated in Figure 3.2. For the DBCM approach we have the following two sub-components:

  1- QoS scaling: the process of converting energy consumption values to energy rating scores in the range of 1 to 5, where 5 correspond to the lowest energy value and 1 corresponds to the highest energy value. The intuition is that service with lower energy

consumption is considered better, thus a higher rating. In order to identify the best service, in the QoS scaling step, the energy rating values need to be generated based on the original energy consumption values.

2- Dataset property conversion: the process of converting dataset properties from numeric values to nominal values in order to use some of the recommendation algorithms. We use this component and the previous QoS scaling component only in the dataset-based contextual modelling approach (DBCM) because in this approach we used a tool (CARSKit [77]) to implement our approach and we added these two components to our proposed system as they are part of the requirements for using the tool and making our datasets match the tool requirements as well.

- **Recommender System:** the component of identifying the best service for the given dataset based on the QoS requirements, in our case, the service that consumes less energy as we focus on recommending energy-efficient web services.
- **Energy measurement framework:** our proposed framework for measuring the energy consumption values associated with the service invocation for a specific dataset.
- **Service hosting server:** the component where all the web services under study are installed and running using a testbed system [54], and all the service invocation requests are sent.
- **Service invocation proxy:** the component that acts as an intermediary for requests from clients invoking services hosted on the server. It is a place where we can record the request details and save them into the log database.

The workflow in our proposed recommendation framework begins by first finding services that match the user's functional requirements. Since we assume that we only deal with clustering

services, we skip the functional matching part. The second step is the non-functional matching that uses QoS properties (the energy consumption values) to prioritize between the services. The goal of this process is to find the best service that matches user requirements and works the best for the input dataset.

To request a service, a user only need to provide the dataset information and then the system will return a recommended list of services based on how good their predicted energy values are for the given dataset (list of energy-efficient services). As we can see in Figure 3.1, at first, a user provides the properties of the input dataset d_t. The system then consults the existing repository to identify datasets similar to d_t. The similarity calculation is based on the dataset properties representing the datasets. The value of per-dataset QoS property can be predicted for the candidate services with the help of the past QoS data of these services on similar datasets. The value of per-service QoS property can be computed as an overall value based on the service performance on all the datasets it has processed. Once the QoS values are estimated for all the candidate services for the input dataset, they can be used for recommending these services. The recommended list of services (S_1, S_2, …, S_n) is then returned to the target user.

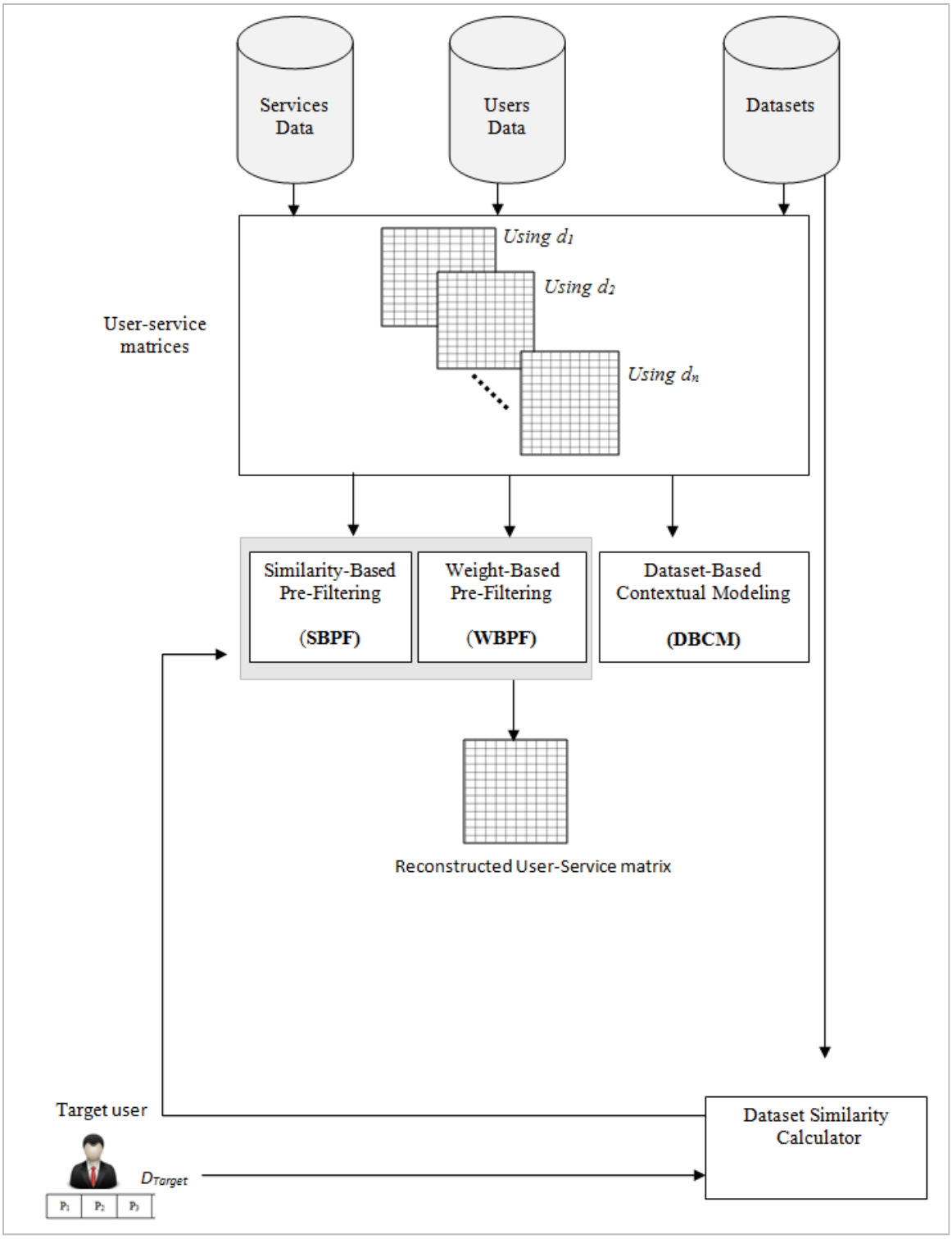

Figure 3.2: Proposed Approaches using Contextual Matrix Factorization.

3.2 Proposed Approaches

Figure 3.2 summarizes the proposed recommendation approaches. The main intention of our proposed recommender system is to recommend energy-efficient data mining web services to the target user by considering the dataset properties. Therefore, the main QoS value that we focus on in this work is the energy consumption value which represents the amount of energy consumed in each invocation of the data mining service to accomplish a specific task. The dataset or the dataset properties, on the other hand, represents the contextual information that we need to integrate into the recommender system.

We proposed different approaches to build our recommender system; they are fundamentally based on the concept of a context-aware recommender system. Since it has been proven in many previous works in the literature that integrating contextual information into the recommendation process can assist in enhancing the prediction and recommendation accuracy, we have decided to use some related contexts in our proposed approaches where the dataset properties are considered as a third dimension and usually this case is not possible in the standard recommender system that deals with only two dimensions. Afterwards, we evaluate the performance of these approaches compared with the baseline models that ignore the contextual information.

In the first approach, similarity-based pre-filtering (SBPF), we use the pre-filtering approach and then we apply the matrix factorization. This approach is explained in detail in section 5.2.1. In the second approach, weight-based pre-filtering (WBPF), we introduce a new method for applying the pre-filtering model which applies weights calculated based on the similarity degree between different contextual factors (dataset properties). This approach is explained in detail in section 5.2.2.

In the last approach, dataset-based contextual modeling (DBCM), we use contextual modeling method to integrate the dataset properties as a context into the recommendation process; this approach is explained in detail in section 5.2.3.

Our proposed recommender system depends on the historical data to predict the missing QoS values. Generally, the historical data in the recommender system might have some missing QoS values, for example; the case of a specific user did not interact with some items/services (in our situation if a user did not use a specific dataset with the previously invoked services) or because some web services haven't been invoked by the user in the past on a given dataset. As a result, our system has the cold-start problem. In order to solve this problem, we first run the system using a set of synthetic datasets and record the corresponding energy consumption values. We want to emphasize that our proposed system is used to predict the missing energy consumption values for the available services based on the used dataset properties. The energy in this case is corresponding to the rating in the standard recommender system.

Chapter 4

Impacts of Different Factors on QoS values

In this chapter, we examine the impacts of different factors such as dataset attributes, service implementations on the amount of energy consumed by each web service invocation as well as the latency levels.

4.1 Experimental Study on Energy Consumption and Latency Behaviour

4.1.1 Datasets

In the field of context-aware recommendation, the available benchmark datasets that can be used for research purposes is very limited [78]. For service recommendation, there are also only a few datasets available such as QWS[2] and WS-DREAM[3]. Our work in this dissertation requires investigating the impact of different dataset properties on QoS values (energy consumption and latency). It is hard to use any of the existing datasets because they did not include the complete information we need to use in our work. Therefore, we decided to use 100 synthetic datasets [5] because the number of real datasets in the known sources (e.g., UCI datasets) is very limited and it is hard to perform any systematic investigation. The synthetic datasets under study differ in their number of attributes, the number of instances, data types and file sizes, which is not easy to find in the existing real datasets.

We conducted many preliminary experiments in order to understand the performance of the used data mining web services. In these preliminary experiments, we used more than 500

[2] http://www.uoguelph.ca/~qmahmoud/qws/
[3] https://github.com/wsdream.

datasets [20] having different sizes and attributes, and we examined the performance in each service invocation. Based on our observations on the initial results, we created our own collection of 11 datasets that show variety in their properties. Table 4.1 shows each dataset with its number of attributes, number of instances, data type and the corresponding file size.

Table 4.1: List of Datasets under Study.

Dataset ID	No. of attributes	No. of instances	Data type	File size (MB)
D_100	100	47263	numeric	40
D_70	70	47263	numeric	27.9
D_40	40	47263	numeric	15.9
D_10	10	47263	numeric	3.81
D_70_Ins	100	33084	numeric	28
D_40_Ins	100	18905	numeric	16
D_10_Ins	100	4726	numeric	4
D_10_DISC		The nominal version of D_10		
D_10_Ins_DISC		The nominal version of D_10_Ins		
D_40_DISC		The nominal version of D_40		
D_40_Ins_DISC		The nominal version of D_40_Ins		

We follow certain rules to assign an ID to each dataset. For example, if the ID has only a number such as D_100 or D_70, then this number represents the number of attributes in that dataset with all available instances (47263); so "D_100" means that the dataset has 100 attributes and 47263 instances. If the ID contains "Ins" (abbreviation for instance), then the number in that ID refers to the percentage of instances with all available attributes (100) in that particular dataset, for example, "D_70_Ins" represents a dataset that has all the available attributes (100) and only 70% of instances (70% of 47263=33084). Finally, if the ID contains "DISC"

(abbreviation of discretize), then it means that this dataset is a nominal version of the original dataset as represented in the ID, for example, "D_10_DISC" is the nominal version of D_10.

Here, D_100 is the base dataset, which has 100 attributes, 47263 instances, and the file size is 40 MB. We used Weka[4] data generator to create it. We then created different groups of subsets of D_100 (the biggest dataset with size 40 MB). The subsets D_70, D_40 and D_10 have the same number of instances as the original dataset, which is 47263, and same data type which is numeric. The different part is the number of attributes. D_70 has 70 attributes, D_40 has 40 attributes and D_10 has 10 attributes.

The subsets D_70_Ins, D_40_Ins, and D_10_Ins have the same number of attributes (100 attributes) and same data type (numeric) as the original dataset. The different part is the number of instances. Each one of these subsets includes a portion of D_100 - 70% (33084 instances), 40% (18905 instances) and 10% (4726) of the original dataset respectively. To generate the nominal datasets, we applied discretize filter from Weka on subsets D_10, D_10_Ins, D_40 and D_40_Ins to get their corresponding nominal versions. Creating subsets of the main dataset is a common method used in the literature [79] to help in creating some datasets that best fit the research problem; we follow this practice in this work.

Then, in order to have a wider variety of datasets, we also used 89 more datasets from the original 500 [20]. So in total, we have 100 datasets. In the preprocessing step, we applied normalization technique on all the datasets attribute values so that they are all in the same range of 0 and 1.

[4] https://www.cs.waikato.ac.nz/ml/weka/

4.1.2 Web Services

Regarding web services, we utilized 10 different data mining services. These services implemented clustering algorithms from three different packages: Weka, R[5] and Apache[6]. The purpose is to examine the behaviour of services implementing different algorithms (same function – clustering) or same algorithm (i.e., k-means clustering) while being implemented and offered by different providers. The algorithms associated with these 10 services (S1-S10) are Simple k-means, Farthest-first, CLOPE, EM, Filtered, X-means from Weka, K-means, hierarchical from R, K-means, DBScan from Apache respectively. Table 4.2 shows the data mining services we have used in our work.

Table 4.2: Data mining services under study.

#	Algorithm	Abbreviation	Implementation
1	Simple k-means	S-Kmeans	Weka
2	Farthest first	FF	
3	CLOPE	CLOPE	
4	EM	EM	
5	Filtered	Filtered	
6	X means	X-means	
7	R k-means	R-Kmeans	R
8	R hierarchical	R-H	
9	Apache k-means	A-Kmeans	Apache
10	Apache DB scan	Adb-scan	

[5] https://www.r-project.org/
[6] https://www.apache.org/

4.1.3 Energy Consumption as a QoS attribute

There are many QoS properties that can be used in the recommendation process such as response time, reliability and availability. In our proposed work, we focus on energy consumption because it reflects the amount of energy required by a web service to accomplish a specific task and thus it is the main factor to determine whether a service is energy-efficient or not.

An Energy-efficient web service consumes less energy to accomplish a task compared with other services; this, in turn, has numerous benefits for various sectors such as the environment, the economy in addition to its main role to push the sustainability process forward. Furthermore, service providers usually try their best to reduce the service's cost, which is directly related to the amount of energy consumed per service. They always have concerns about the amount of energy consumed by their services as it affects the operational cost and they prefer to run more energy-efficient services. Hence, our presented work has potential impacts and benefits to these providers and denotes an important contribution that can save thousands of dollars by cutting down the amount of energy consumption. The service providers can calculate and record the energy consumption values while the service is being invoked. Therefore, there should be no extra cost for this calculation. In other words, our main intention here is to recommend an energy-efficient service, and in case some services have similar functionality and similar quality levels, consumers usually would not mind being offered an energy-efficient service.

In the literature, there are many approaches for considering multiple QoS properties [80]; therefore, we can easily extend our current work using those existing approaches to include multiple QoS values.

4.1.4 Used Machine

The machine we used in all the experiments is AMD Turion ™ 64 x 2 Mobile Technology TL-60 2.00 GHz with 3GB memory. All the experiments have been run using the following settings on the computer: 1) 100% fully charged battery while connecting to the power meter; 2) display settings changed to “do not dim the display”, “do not turn off the display” and “do not put the computer to sleep”. The purpose is to ensure that the energy measurements will not be affected by any of the display settings.

In order to avoid measurement errors and ensure the accuracy of the experiment results, initially, we repeated each test five times to get the average value. We found that some web services took a long time when processing large datasets. Also, we found that the readings we get from each experiment run (for the same service) were very close to each other, therefore, we decided to run each test two times and then calculate the average of the readings.

4.2 Examining the Impacts of Different Factors on QoS Values

In this part, we conducted various experiments and our main goal is to examine the factors that might affect the amount of energy associated with every single invocation of data mining web service when utilizing a specific dataset in addition to examine the behaviour of the latency in these invocations. We mainly focus on examining the effect of dataset properties and service implementations.

4.2.1 Impact of Data Type

In this set of experiments, we study the energy consumption and latency behavior of web services implementing clustering algorithms as listed in Table 4.2. We ran all services over the following datasets: D_10, D_10_Ins, D_40, D_40_Ins, and their corresponding nominal

versions. We did the pair-wise comparison between datasets that have the same number of attributes and instances while different data types. For example, we compared the measurements from D_10 and D_10_DISC, D_40 and D_40_DISC, and so on. We investigated the impact of data types of the dataset on energy consumption and latency for each web service.

Most of the services were successfully run on datasets with both data types: numeric and nominal, except Weka X-means and Apache K-means, as these two services work only with numeric datasets and cannot handle nominal data. In the case of R k-means, both data types are working because R comes with a specific distance function (e.g., Gower) for nominal data.

We used equation (4.1) to calculate the difference between each pair of two measurements: measurement on the numeric dataset and measurement on the corresponding nominal version. Since in this experiment we want to investigate if the data type affects the amount of energy consumption of the invoked service (or latency behaviour), we decided to get the absolute value of the "percentage of difference" results so we can focus on the amount of changes regardless it is an increase or decrease.

$$\text{Percentage of difference} = \left|\frac{(\text{M1} - \text{M2})}{\text{M1}}\right| \times 100 \tag{4.1}$$

where *M1* and *M2* represent the measurements of latency or energy values on the numeric dataset and nominal version of the same dataset respectively. Figure 4.1 and Figure 4.2 show the differences in energy and latency correspondingly, the results in these two figures illustrate how energy consumption and latency are affected by data types.

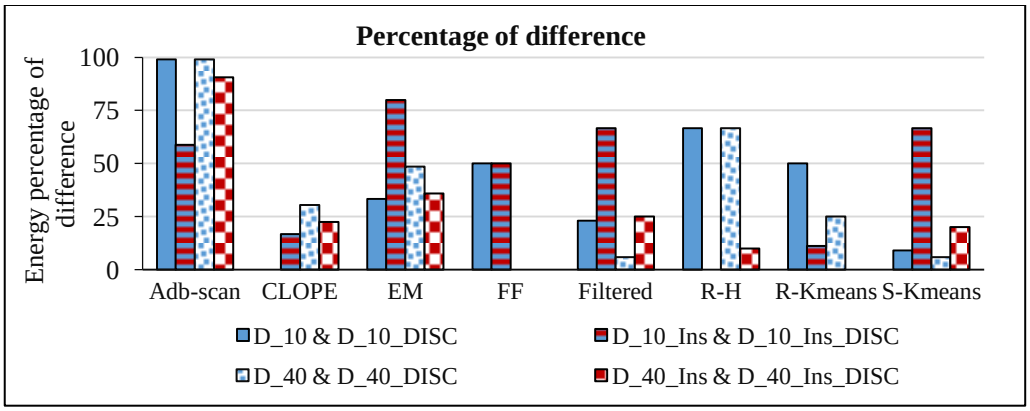

Figure 4.1: Energy percentage of difference of 8 web services using 8 datasets (absolute values) [5].

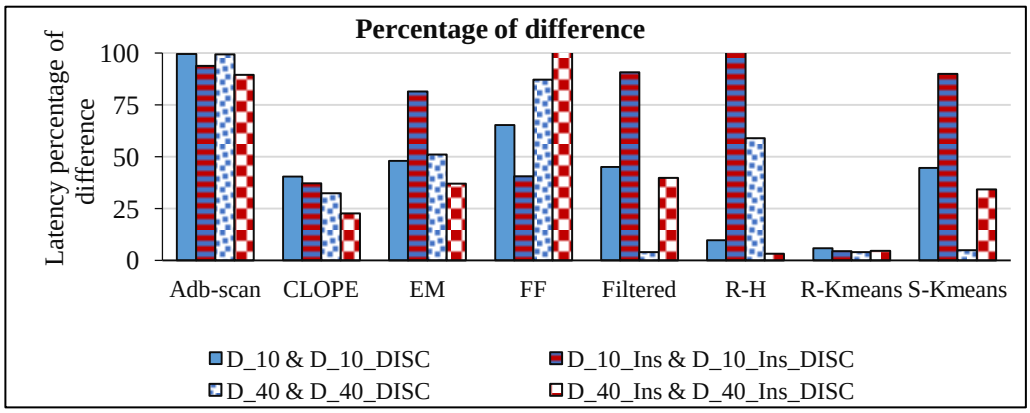

Figure 4.2: Latency percentage of difference of 8 web services using 8 datasets (absolute values) [5].

As we can see in Figure 4.1, most of the services have different energy consumption values for numeric and nominal datasets. The difference could be as high as 100% or as low as 0%. Services such as Adb-scan, EM, or Filtered consume more energy on numeric datasets than that on nominal datasets. Services such as R-K-means or FF consume less energy on numeric datasets than that on nominal datasets. There are also services such as CLOPE that have mixed results. Also, we can observe a smaller difference for some services such as CLOPE.

The latency results show similar outcomes as we can see in Figure 4.2. However, the scale and pattern of the difference is not the same for latency as compared to the ones for energy. For instance, the scale of energy difference of Adb-scan on 4 datasets is between 60% and 100% whereas, the scale of latency difference is between 90% and 100%. For R-H (R hierarchical), it has higher energy value on D_40_ DISC (a nominal dataset) whereas lower latency on the same dataset. In a way, it indicates that energy and latency show different patterns on different datasets.

To summarise the results from this set of experiments, we can conclude that the energy consumption of web services, as well as the latency, can be affected by the data type. Some services consume less energy when processing datasets with numeric data types compared to the case when processing nominal counterparts. Some services consume more energy on numeric data. In general, services consuming more energy on a certain dataset also have higher latency values, although this pattern is not consistently shown in all cases.

4.2.2 Impact of Number of attributes

In this set of experiments, we examine the impact of the number of attributes of the dataset on energy consumption and latency. For this purpose, we used the original dataset D_100 together with its three subsets: D_70, D_40 and D_10. We ran the experiments by invoking all the services on these four datasets. The corresponding energy consumption and latency values were recorded for each service. Figure 4.3 and Figure 4.4 demonstrate the energy consumption and latency behaviour of ten clustering services on these four datasets.

By examining Figure 4.3 and Figure 4.4, we can see that increasing number of attributes leads to an increase in latency values for most of the services, except for Adb-scan, which does

not have an obvious difference in its latency values. For the energy results, we found that for most of the services, the change in the number of attributes does not have a big impact on the energy values. For two services CLOPE and EM, energy is increased when the number of attributes increases. For Adb-scan, it is not strictly increasing. For all the other services, there is only a slight increase in energy values when the number of attributes increases.

The outcome from this set of experiments shows that the number of attributes has less impact on energy than latency. It also indicates the difference between latency and energy and how they are affected by the same parameter differently.

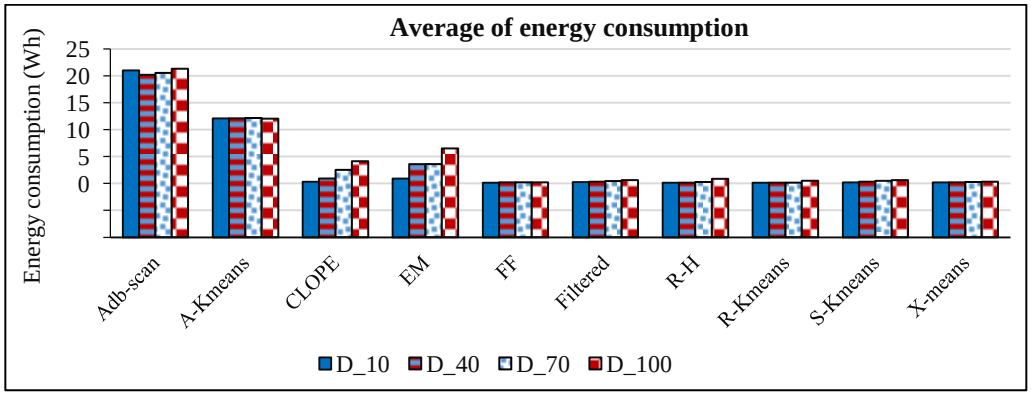

Figure 4.3: Average of energy consumption for web services on four datasets with a different number of attributes [5].

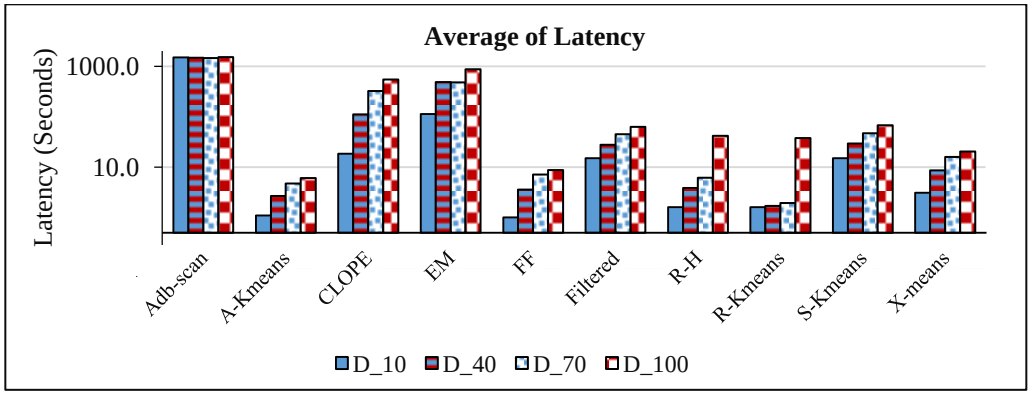

Figure 4.4: Average of latency for web services on four datasets with a different number of attributes, (using log scale) [5].

We calculated the percentage of difference for both energy and latency between D_100 and D_10 by using the equation shown in (4.1) and the results are shown in Figure 4.5. The results show the different impact on energy and latency. In general, latency is affected more by the number of attributes than energy, and Adb-scan is not affected much by this dataset attribute. Figure 4.3 and Figure 4.4 are scaled to show the small values clearly.

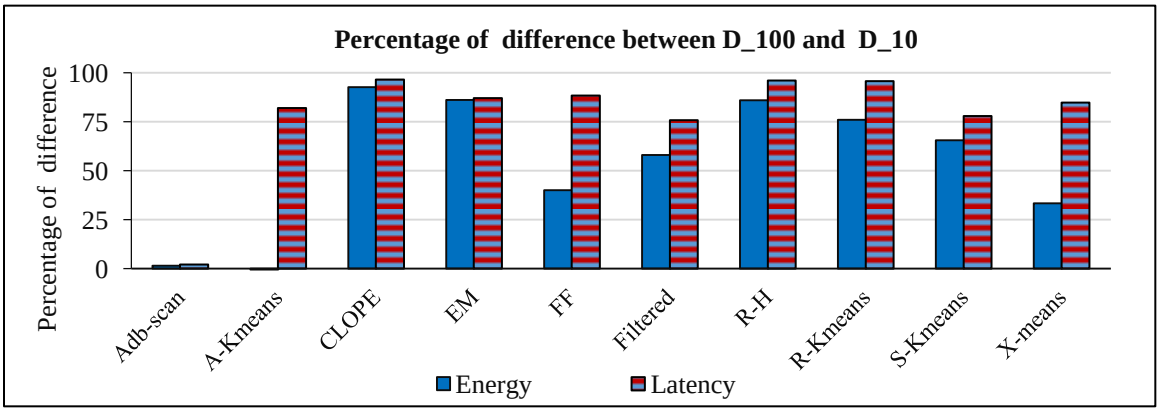

Figure 4.5: Percentage of difference between D_100 and D_10 for energy consumption and latency.

4.2.3 Impact of Number of Instances

In this set of experiments, we examine the impact of the number of instances of the dataset on energy consumption and latency. For this purpose, we used the original dataset D_100 together with its subsets D_70_Ins, D_40_Ins, and D_10_Ins. Figure 4.6 and Figure 4.7 show the impact of the number of instances on energy consumption and latency on these four datasets when the number of instances changes.

By examining Figure 4.6 and Figure 4.7, we can see that when the number of instances increases, latency values increase for most of the services, except that for a few services (e.g., EM and Filtered) the latency is not strictly increasing. For the energy results, 3 services (Adb-scan, A-Kmeans and CLOPE) exhibit a clear increasing pattern in energy consumption as a

result of the increasing number of instances. EM also shows a clear increase in its energy values, although it is not strictly increasing. For the other services, the change in the number of instances does not have a big impact on energy values. There is only a slight increase in their energy values. This result shows that the number of instances has less impact on energy than latency. It also indicates the difference between latency and energy shows that we need to consider latency and energy separately.

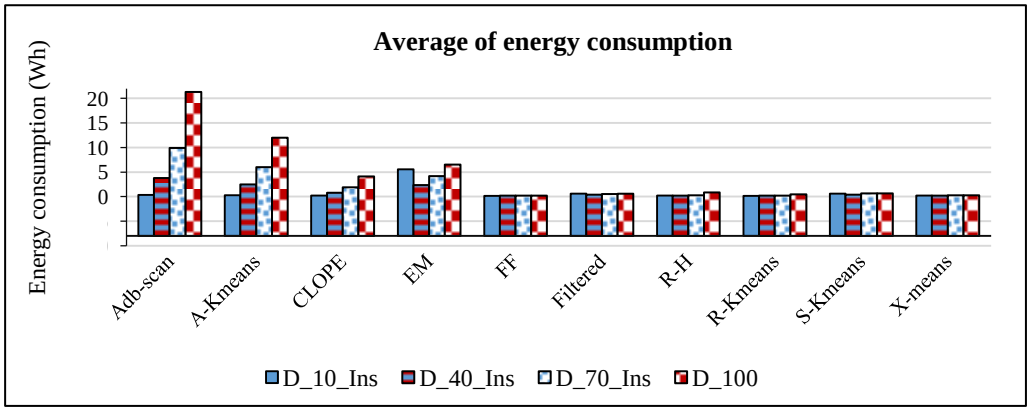

Figure 4.6: Average of energy consumption for web services on four datasets with a different number of instances [5].

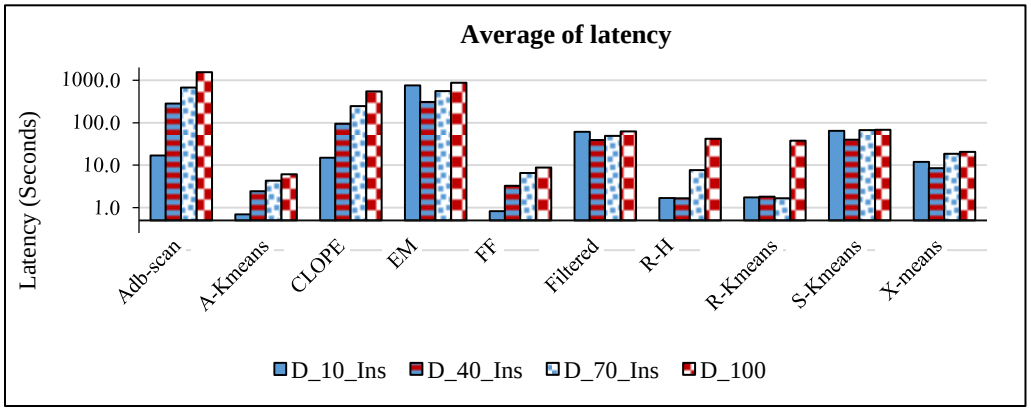

Figure 4.7: Average of latency for web services on four datasets with a different number of instances, (using log scale) [5].

We calculated the percentage of difference for both energy and latency between D_100 and D_10_Ins using equation (4.1) and the results are shown in Figure 4.8. The results show the different impact on energy and latency. The difference between energy and latency for seven services is very clear whereas, for the other three services the difference is very little. Figure 4.6 and Figure 4.7 are scaled to show the small values clearly.

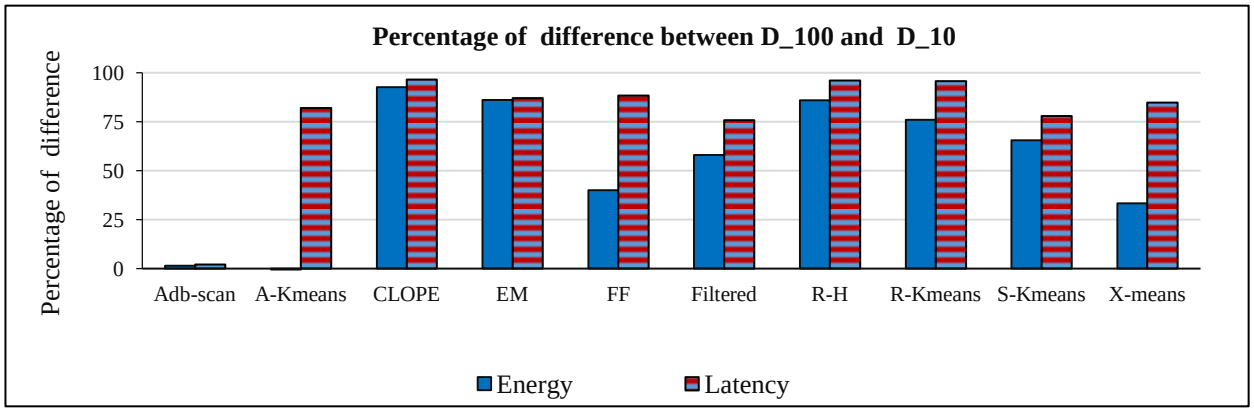

Figure 4.8: Percentage of difference between D_100 and D_10_Ins for both energy and latency.

4.2.4 Impact of Using Different Implementations

In this set of experiments, we study the impact of different implementations of the same algorithm and we examine how different implementations might affect the energy consumption and latency. For this purpose, we used three web services from three different implementations: Apache K-means, R K-means and Weka simple K-means. They are all implementations of the same algorithm (k-means) and they are from Apache, R and Weka respectively. We invoked these three services on 7 datasets and the results are shown in Figure 4.9 and Figure 4.10.

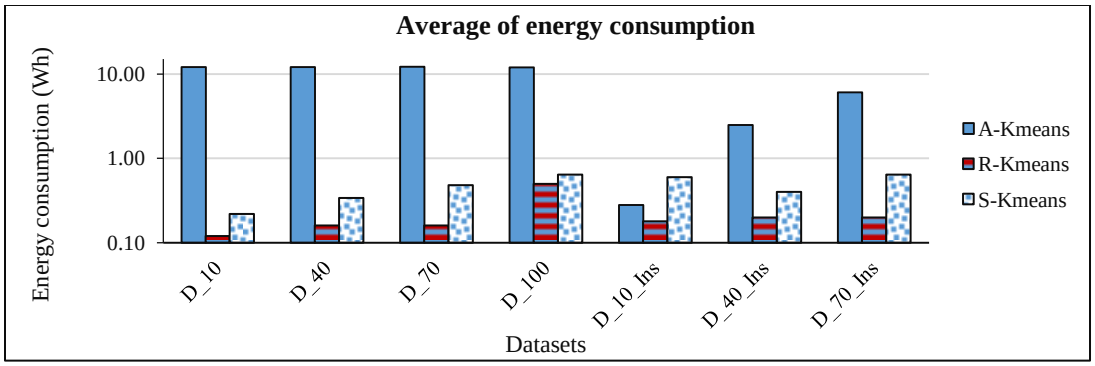

Figure 4.9: Average of energy consumption for three web services with different implementations of the same algorithm on seven datasets, (using log scale) [5].

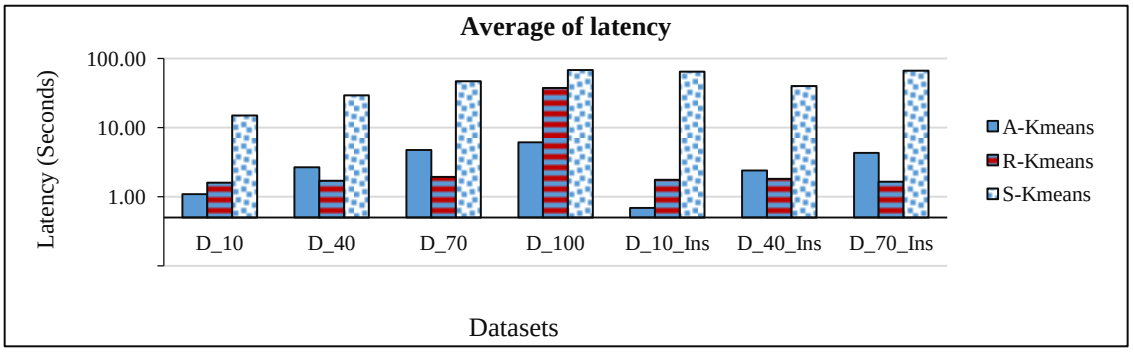

Figure 4.10: Average of latency for three web services with different implementations of the same algorithm on seven datasets, (using log scale) [5].

As we can see from Figure 4.9, energy consumption is affected by how web services are implemented. We found that in general, Apache k-means consumes more energy than the other two implementations, and R k-means consumes the least energy among the three implementations. For the latency, as shown in Figure 4.10, Weka S-Kmeans has the highest latency among three services, and between Apache K-means and R K-means, there is no clear winner, for some datasets, Apache K-means has a lower latency, for others, it has a higher latency.

The result shows that using different service implementations can have a significant influence on energy consumption levels and latency. Moreover, the outcomes from this set of experiments show that the latency follows a different pattern than energy consumption.

Our justification for this result is that having different implementations means using different ways of coding, which might affect the used resources and consequently affect the QoS values such as the amount of energy consumption.

In this dissertation, our focus is on using different services implementing the same functionality (clustering) and not how the services are implemented or the used algorithm. Since the outcome we gained from this experiment shows that the service implementation might be one of the factors that can affect the energy consumption levels of the invoked services, we plan to add the service implementation as a context information in our future work. Further, our focus in this dissertation is on the energy consumption values as it represents the standard to determine the lowest energy consumption service.

4.3 Summary

In this chapter, we have implemented an experimental study on energy consumption and latency behaviour. We examined the amount of energy associated with each single web service invocation as well as the latency values when using different datasets. We discovered that the latency and energy consumption of web services can be affected by different factors in relating to the used dataset such as the data type, number of attributes and number of instances. Also, we discovered that the amount of energy and the latency can be affected by service implementations.

Moreover, we found that some factors have more clear impacts on energy levels (latency) than the others. For example, using services from various implementations can have a significant

influence on energy consumption, while the number of attributes in the dataset can have less impact on energy values. In the later part of this dissertation, we only focus on dataset properties as contextual factors, without considering the impact of service implementations. Additionally, we focus on energy consumption as the main QoS property for recommending energy-efficient service.

Chapter 5

Methodology:
Context-aware Recommendation Approaches for Energy-Efficient Data Mining Web Services

In the proposed recommender system we introduced three approaches: similarity-based pre-filtering (SBPF), weight-based pre-filtering (WBPF), and dataset-based contextual modeling (DBCM). In the first approach, we applied the pre-filtering technique in order to be able to use the third dimension (dataset) and change the recommendation problem from three dimensions to two dimensions, then we applied matrix factorization model for predicting the missing energy values. In the second approach (WBPF), we introduced a new method for incorporating the third dimension into the recommendation process, where we consider the similarity of the contexts as a weighted factor for adjusting the rating values before applying the pre-filtering technique. In the last approach, we applied contextual modeling to incorporate the dataset as contextual information. We compared the three proposed approaches in terms of prediction accuracy and recommendation accuracy and we made our suggestions on when to use which one.

5.1 Preliminaries

5.1.1 Similarity Measurement of the Dataset Properties

In order to measure the similarity between the properties of any two datasets, first we represent the properties of each dataset (number of instances, number of attributes, data type and file size) as one vector that contains four elements and each element refers to a single property. Second, we compare each vector (representing one dataset) with the remaining vectors

(representing remaining datasets) to find the similar ones by using cosine similarity method as shown in equation (5.1).

The reason we chose cosine similarity among the other available methods is that it is known as one of the common methods used in measuring the similarity between two non-zero vectors. Also, cosine similarity is well known as a very efficient method, especially for sparse vectors, as only the non-zero dimensions need to be considered. Additionally, cosine similarity is normally used in positive space, where the outcome is neatly bounded in [0, 1]. Furthermore, in our preliminary test, we tried other methods such as Euclidean distance but we found that the results we obtained from cosine measurements are more accurate. For all these reasons we decided to use cosine to measure similarity degree rather than other existing methods.

First, we normalize all the data related to dataset properties so that all the values range between 0 and 1. Then, we find the similarity degree between each pair of datasets. The value would be 1 in case of matching vectors and 0 when the two vectors do not share any attributes which is the case of dissimilarity. The cosine similarity formula is defined as shown in equation (5.1).

$$Sim(A, B) = Cos(\theta) = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}} \tag{5.1}$$

where *A* and *B* are two different datasets having the same number of attributes (*n)*, A_i and B_i represent the corresponding attributes from datasets *A* and *B* respectively.

5.1.2 Matrix Factorization

Matrix Factorization is one of the most effective algorithms for implementing recommender systems. Both user-related and item-related latent factors are represented as vectors and the rating prediction can be defined as the product of these two vectors. In our proposed work, items are services and ratings are QoS values (energy) experienced by users on those services. We collected the QoS values resulting from each user request to any one of the existing services, and we are focusing on predicting the missing QoS values of web services which have not been invoked by users as shown in equation (5.2).

$$\hat{Q}_{us} = U \times S \tag{5.2}$$

The values in the user vector *U* and service vector *S* indicate the weights on *k* latent factors. The weights in the service vector represent how much (or to what extent) the service is associated with those latent factors, and the weights in the user vector represent how much users like those latent factors. As a result, the dot product of these two vectors can be used to indicate the QoS value of a service invoked by a user. Furthermore, user and service biases can be added to the prediction function, as shown in equation (5.3) because it is proved that many recommender systems perform better if the user and item biases are taken into account [66].

$$\hat{Q}_{us} = U \times S + b_u + b_s + \mu \tag{5.3}$$

where μ represents the overall QoS average in the data, b_u and b_s represent the user and service bias which reflect the observed deviations of user *u* and service *s* from the average respectively. Therefore, the predicted QoS value consists of four elements: user-service interaction, user bias, service bias, and overall average.

The recommender system learns user and service vectors, user bias and service bias by Stochastic Gradient Descent (SGD) method to minimize the QoS prediction errors (by minimizing the squared error function) as shown in equation (5.4).

$$min_{U^*,S^*,b^*} \sum_{(u,s)\in K} \left[(Q_{us} - U \times S - b_u - b_s - \mu)^2 + \lambda(b_u^2 + b_s^2 + \|U\|^2 + \|S\|^2\right] \quad (5.4)$$

where K is the set of the (u, s) pairs for which Q_{us} is known, and λ represents the regularization factor.

5.2 Proposed Recommendation Approaches

In general, the main goal of a recommender system is to recommend an item or service to the target user according to the user's preferences. In context-aware recommender system, the recommendation goes beyond the user's preferences and takes into consideration the contextual situation of the users while generating the recommendations.

In our proposed approaches, the datasets and their associated properties represent the context that we need to integrate into the recommender system seeking for the energy-efficient data mining services. Each dataset has a set of properties which is used to distinguish one from another such as the number of instances, data type and the number of attributes. Since there is a strong relation between dataset properties and the QoS properties of data mining web services (e.g., energy and latency) as shown in chapter 4, we revealed that there is a clear effect that can be noticed – when there are changes in the dataset properties, the QoS values (energy consumption) of data mining web services are changed as well. Therefore, in our proposed approaches we introduce data as a new contextual factor that can be added to the existing web service recommender system for the sake of improving both prediction and recommendation

accuracy. In our proposed approaches, we consider a case of recommending a data mining service for a user who has a specific dataset and has not invoked such service before.

First, we start by predicting the missing QoS values represented by energy consumption; we implement that based on the historical data of the similar datasets used by either the same user or users who are similar to the target user. Afterwards, a list of candidate services along with their associated predicted QoS values are constructed. Finally, the service that shows the acceptable QoS values (energy consumption) is recommended to the target user as the energy-efficient data mining service.

5.2.1 Similarity-Based Pre-Filtering (SBPF)

In this proposed approach, we use the pre-filtering method because it allows the deployment of any of the traditional recommender system techniques previously proposed in the literature in addition to its easy implementation [15]. In the pre-filtering, the contextual information serves as a query for filtering relevant rating data only. By applying pre-filtering, we can reduce multiple dimensions into two dimensions and then all the research on two-dimensional recommender system is directly applicable.

In this approach, we can do the exact pre-filtering, which might give a very narrow result, so it may not have enough data for accurate QoS prediction (known as sparsity problem). Therefore, it would be better to use more general context specification than using the exact context.

Generally, in CARS, for given dimensions D_1, D_2,…, D_n, the rating function is defined as shown in equation (5.5).

$$R: D_1 \times \ldots \times D_n \rightarrow Rating \tag{5.5}$$

Accordingly, the three-dimensional QoS prediction function supporting data as the context can be defined as shown in (5.6).

$$\hat{Q}^{V}_{User\times Service\times Dataset} : U \times S \times D \rightarrow QoS \tag{5.6}$$

where *V* contains a set of records (user, service, dataset, QoS).

By applying the pre-filtering method, the contextual information (dataset properties) is used to filter out irrelevant information before they are utilized for computing recommendations via standard recommender system method. In this work, we use the information of the current (target) dataset to filter out the unrelated datasets. A three-dimensional prediction function (with exact pre-filtering) can be stated through a two-dimensional prediction function as shown in equation (5.7):

$$\begin{aligned} &\forall (u,s,d) \in U \times S \times D, \\ &\hat{Q}^{V}_{User\times Service\times Dataset}(u,s,d) = \hat{Q}^{V[dataset=d_{Target}](User,Service,QoS)}_{User\times Service}(u,s) \end{aligned} \tag{5.7}$$

where $[dataset=d_{Target}]$ and $V[Dataset=d_{Target}](User,Service,QoS)$ represent respectively, a simple contextual pre-filter and a QoS dataset obtained from *V* by selecting only records where dataset dimension has value d_{Target} and keeping only the values for *User* and *Service* dimensions, as well as the value of *QoS* itself (in this case the energy value).

In other words, if we consider a dataset of three-dimensional values *V* as a relation, then $V[Dataset=d_{Target}]$ *(User, Service, QoS)* is another relation obtained from *V* by performing two relational operations: selection (selecting similar datasets) and projection (excluding the data dimension).

To have a more general context specification, we use similarity function to construct a subset of datasets including all datasets similar to the target one rather than using the exact dataset. Afterwards, we can predict the QoS by using the equation (5.8):

$$\hat{Q}^{V}_{User\times Service\times Dataset=}\hat{Q}^{V[Dataset\in C_d](User,Service,AVG(QoS))}_{User\times Service}(u,s) \tag{5.8}$$

$$C_d = \{dataset|\ Sim\big(dataset \in DS\ , d_{Target}\big) > t\} \tag{5.9}$$

where *DS* represents the set of all existing datasets in the dataset repository, C_d represents a subset of datasets that include all datasets satisfying the condition shown in equation (5.9), and *AVG (QoS)* represents the average function for the QoS values, given that there might be many QoS values with the same *User* and *Service* for different dataset instances in dataset *V* belonging to some contextual combination *Cd*.

For example, if $dataset_1$ and $dataset_2$ are both belonging to the same subset; they might have different QoS values. Therefore, we aggregate these values by using the average function as illustrated in equation (5.8). Equation (5.9) shows the similarity function $Sim(dataset,d_{Target})$ which is used to find a dataset *dataset* having properties similar to the target dataset d_{Target} and *t* represents a threshold value which we can set.

The idea of the pre-filtering technique is applied before in some other previous works [15]. In this dissertation, we introduce to add new contextual information represented by the data properties to be used in the pre-filtering process. We started our implementation for this technique by first calculating the similarity degree between the target dataset and all the

remaining datasets in the dataset repository (as shown in section 5.1.1) and then we select only the datasets those give similarity degree bigger than the specified threshold.

We have tried different values for the threshold and examined the corresponding Root-Mean-Squared-Error (RMSE) between the original energy values and the predicted values. Based on our experiment results, we chose the threshold value as 0.5. Choosing 0 as the threshold value means retrieving all existing datasets whereas choosing 1 implies retrieving very few datasets. A middle value 0.5 would be a trade-off between the two extremes. After selecting the datasets that show similarity degree bigger than 0.5 (by using cosine similarity as explained in section 5.1.1 in this dissertation), we calculate the average of the energy values for those selected datasets for each pair of user and service. In the next step, we applied the matrix factorization technique for predicting any energy missing values as illustrated in equation (5.8).

Our contribution in regard to this approach can be represented by the new dimension (dataset) that we introduced for implementing the filtering process. Accordingly, we select similar datasets based on their properties and finally aggregate the associated QoS properties from those datasets. In the final steps the missing QoS values (energy) are predicted, the recommendation list of candidate services is constructed and consequently, services with the lowest amount of energy are recommended to the active user.

5.2.2 Weight-Based Pre-Filtering (WBPF)

In this approach, we propose to use a new method for implementing the pre-filtering technique. The proposed method focuses on using weighted factors measured based on the similarity degree between the target dataset and all the remaining dataset in the dataset repository. Initially, we have the following information from each service invocation:

user, service, energy (rating), dataset ID, and dataset properties. We represent the dataset properties as vector and accordingly we can use this vector to measure the similarity between the target dataset and the other existing datasets in our system. Whenever a new dataset is presented in the system and used by one of the users, this dataset with its properties would be added into the dataset repository. If we assume a target user has a dataset (d_{Target}) and looking for a data mining service for a specific task, in this proposed approach the sequence of the steps are shown below:

First, we calculate the similarity degree between the target dataset d_{Target} and the remaining datasets in our dataset repository using cosine similarity formula as shown in equation (5.1). After finding the similarity degree, for each (user, service, d_{Target}) we should obtain the following results:

User, service, energy, d_{Target}, d_1, $\text{Similarity}(d_{\text{Target}}, d_1)$
User, service, energy, d_{Target}, d_2, $\text{Similarity}(d_{\text{Target}}, d_2)$
User, service, energy, d_{Target}, d_3, $\text{Similarity}(d_{\text{Target}}, d_3)$
.
.
.
User, service, energy, d_{Target}, d_n, $\text{Similarity}(d_{\text{Target}}, d_n)$

Where n represents the number of available datasets in the dataset repository, $Similarity(d_{Target}, d_i)$ represents the similarity degree between d_{Target} and d_i, and i is in the range of 1 to n.

In the second step, we consider each of the similarity values as a weight variable that can have an impact on the energy values depending on how similar the target dataset is, compared

with the remaining datasets. So, in this step, we calculate the new raring $Rating_{new}$ via multiplying the original rating by the similarity degree as shown in equation (5.10):

$$\mathrm{Rating_{new}} = \mathrm{Rating_{Original}} \times \mathrm{Similarity}(\mathrm{d_{Target}}, \mathrm{d_i}) \quad (5.10)$$

Where $Rating_{Original}$ represents the original energy value obtained from invoking web service by the target user and using dataset d_i, $Similarity(d_{Target}, di)$ is the similarity degree between d_{Target} and d_i, and d_i is a dataset in the dataset repository.

So from this step, we would have a new record for every single invocation as the following:

User, Service, $\mathrm{Rating_{Original}}$, d_{Target}, $\mathrm{d_i}$, $\mathrm{Similarity}(\mathrm{d_{Target}}, \mathrm{d_i})$, $(\mathrm{Rating_{Original}} \times \mathrm{Similarity}(\mathrm{d_{Target}}, \mathrm{d_i}))$

After that, we calculate the average for each (user, service, dataset) set as shown in equation (5.11):

$$\frac{\left(\sum_{\mathrm{i=1}}^{\mathrm{N}} \mathrm{Sim}(\mathrm{c_{Target}}, \mathrm{c_i}) \times \mathrm{R_{usc_i}}\right)}{\mathrm{N}} \quad (5.11)$$

Where c_{Target} and c_i represent the *target* context and i^{th} context respectively, R_{usc_i} represents the rating obtained when user *u* invoked service *s* and using c_i context and *N* represents the total number of contexts (datasets in our case).

More specifically, for each (user, service, $\mathrm{d_{Target}}$), we calculate the average rating (average of energy values) as shown in equation (5.12).

$$\frac{\left(\sum_{\mathrm{i=1}}^{\mathrm{N}} \mathrm{Sim}(\mathrm{d_{Target}}, \mathrm{d_i}) \times \mathrm{energy_{usd_i}}\right)}{\mathrm{N}} \quad (5.12)$$

Where d_{Target} and d_i represent the *target* dataset and i^{th} dataset respectively, $energy_{usd_i}$ represents the energy consumed as a result of invoking service *s* by user *u* and using dataset d_i , and *N* represents the total number of datasets in the dataset repository.

In case there are no dataset properties or we could not calculate them, we just consider the rating values without the similarity.

In the last step in this approach, the matrix factorization is used to predict any missing QoS values and after that, the recommendation list of the optimal services is constructed. The services that exhibit the lowest amount of energy consumption are considered as the energy-efficient data mining web services which can be recommended to the target user.

5.2.3 Dataset-Based Contextual Modeling (DBCM)

Contextual modeling incorporates context directly into its recommendation process. In the basic recommender system, we usually have two-dimensional data – user and item. The context under which user and item interact can be included alongside the user and item data. There could also be multiple contextual factors.

Predictive models such as context-aware matrix factorization, regression and decision trees are examples of contextual modeling techniques. Context-aware recommender systems turn the prediction task into a multidimensional rating function R: Users × Items × Contexts → Ratings [81]. The contextual modeling approach has been shown to be able to improve the rating prediction accuracy and produce a better context-aware recommendation than the filtering-based approaches [78]. In our previous approach introduced in section 5.2.1 and 5.2.2, we have used the pre-filtering approach to incorporate various dataset properties as the contextual information into the matrix factorization model to predict the missing QoS values (energy) and recommend the energy-efficient web services.

In this approach, we decided to use the context-aware matrix factorization model [82, 83] through the adoption of contextual modeling technique to build our recommender system. In

order to investigate which dataset related information can give a better recommendation result, we explore different methods of representing and combining the dataset information as contextual information. This part represents our contribution with regard to this technique.

In the existing works in the area of a context-aware recommender system, usually, they consider the situation of having contextual information to implement the recommendation without examining the effects of using different combinations of this information and how it might affect the recommendation accuracy. For example, using the weather information as contextual information in recommending a place of interest to the target user or depending on the companion information for recommending a music track or movie to the user [18][63].

To the best of our knowledge, no investigations of the various methods of representing and integrating this contextual information have been done in this research area. Therefore, in this approach, we decided to do this kind of investigations. Additionally, in this approach, we investigate the impact of using different combinations of the available contexts (dataset properties in our case) and we demonstrate how doing this type of representation can affect the recommendation process.

In order to represent dataset information, we adopt two different methods. The first method is to consider the dataset itself as the contextual dimension. In this case, the contextual dimension is the dataset, and each unique dataset represented by its ID is the contextual condition. The second method is to treat each dataset property as a contextual dimension. In this approach, we consider three dataset properties: number of attributes (P1), number of instances (P2) and data type (P3). According to our experimental results, as presented in Chapter 4 of this dissertation, they all have an impact on the energy consumption values of data mining services.

Our approach is not restricted to these presented properties. Other dataset properties that may potentially affect the energy values can also be added into the model.

By using the first method of representing the context (as mentioned in this section), we will test different dataset properties and their combinations to check which one can give the best recommendation result. Between these two representation methods, the former one is more suitable if there are not many datasets and each one has been used in a number of service invocations with different services applied. In this case, there are not too many contextual conditions for this contextual dimension, and each context condition has enough historical data. However, if there are many datasets or each dataset is only used once or twice in the past invocations, this method may not work well. When there are many datasets or a limited amount of historical data for each dataset, the latter method might be a better option.

When using dataset properties, for numeric properties such as the number of instances, we discretize the numeric values and convert them to nominal values. The reason is that the exact difference between the two property values is not that important. For example, a dataset with 50 instances and a dataset with 70 instances, the absolute difference is 20; however, data mining algorithms could perform very similarly on the two datasets. There is usually no obvious difference in terms of performance or energy consumptions. In this case, using a range (or interval) to represent the dataset property is more appropriate, with each range representing a certain type of dataset. For example, datasets with less than 10k instances will achieve similar performance using a certain data mining algorithm. For this reason, we convert these numeric dataset property values using the conversion rules defined in Table 5.1. This is also the requirement of the tool we used for the experiment.

Table 5.1: Converting the numeric dataset property values.

P1 (no. of attributes)		P2 (no. of instances)	
Range	**Nominal value**	**Range**	**Nominal value**
10-29	X-Small	1-12333	X-Small
30-49	Small	12334-23833	Small
50-69	Medium	23834-35333	Medium
70-89	Large	35334-46833	Large
90 +	X-Large	46834-58333	X-Large

In CARS, the contextual dimensions usually represent the attributes of the activity itself while the user or item attributes are considered as content in recommender systems. Some typical examples of the context information that can be integrated into the recommendation model include time, location, companion. This information represents the contextual dimensions that each of them can have different values representing the contextual conditions.

Generally, energy values can be any real numbers in Watt per hour, representing the amount of energy required when a user invokes a web service to accomplish a task on a specific dataset. Since in our proposed approach the main intention is to recommend the web service that consumes less energy, we scale the energy values so that the lowest energy value is corresponding to 5 (as a rating value) while the highest energy value corresponds to 1. This is also the required input for the tool we used in the experiment. According to the actual energy values we collected in our experiment, we define the following scaling rules: if the energy value ≤ 0.1, the energy rating is set to 5; else if energy ≤ 0.15, rating is 4; else if energy ≤ 0.2, rating is 3; else if energy ≤ 2.5, rating is 2; else, rating is 1.

In this approach, we did not consider the file size property because we discover that this property has a dependency on the other properties. In other words, having more dataset attributes

or instances or using a specific data type can lead to having a dataset with big file size. Therefore, we decided to not consider the file size in this approach.

The contextual modeling approach is used in many previous works [12] [18] and has been proved to have a better recommendation result comparing with the traditional recommendation approach. In our work, we propose to add dataset properties as contextual information into this approach and investigate whether this incorporation might improve the web service recommendation accuracy.

In this work, we mainly focus on two dependent contextual modeling approaches similarity-based CAMF and deviation-based CAMF models as explained in detail in section 2.1.2 in this dissertation. According to our test results, CAMF_LCS (similarity-based) and CAMF_CI (deviation-based) are the best performing models from each category respectively. Therefore, we chose these two as representative models. We refer to CAMF_LCS model as SIM and to CAMF_CI model as DEV in the rest of this dissertation. The prediction formula for DEV model is shown in equation (5.13) [19].

$$\hat{Q}_{usc_1c_2\ldots c_N} = \overrightarrow{p_u} \cdot \overrightarrow{q_s} + b_u + \sum_{i=1}^{N} CQD(c_i, s) + \mu \quad (5.13)$$

where $CQD(c_i, s)$ represents the contextual QoS deviation associated with service s and the contextual condition in the i^{th} dimension. This approach assumes that the contextual QoS deviation is dependent on services, therefore, it replaces b_s in equation (5.3) by the contextual QoS deviation term.

In general the prediction process of similarity-based CAMF model (SIM) is shown in (5.14) [12],

$$\hat{Q}_{usc_k} = \overrightarrow{p_u} \cdot \overrightarrow{q_s} \cdot Sim(c_k, c_E) \quad (5.14)$$

where c_E represents the empty context situation (the value in each contextual dimension is empty or "NA"), c_k represents a non-empty contextual situation. The function $Sim(c_k, c_m)$ estimates the correlation between c_k and c_m where at least one contextual condition is not empty as presented in equation (5.15) [12].

$$Sim(c_k, c_m) = \prod_{l=1}^{L} similarity(c_{k,l}, c_{m,l}) \tag{5.15}$$

where c_k and c_m are two contextual situations, and $c_{k,l}$ represents the l^{th} context condition. In CAMF-LCS, this similarity function is defined in equation (5.16) [12].

$$similarity(c_{k,l}, c_{m,l}) = V_{c_{k,l}} \cdot V_{c_{m,l}} \tag{5.16}$$

where $V_{c_{k,l}}$ and $V_{c_{m,l}}$ represent the vector representation for contextual condition $c_{k,l}$ and $c_{m,l}$ respectively over the space of latent factors.

5.3 Summary

In this chapter, we have presented the methodology we followed in implementing our three proposed approaches towards building an energy-efficient web service recommender system. Additionally, we introduced the method we followed to measure the similarity degree between datasets. We explained in detail each of the proposed approaches to highlight the differences among them and also to clarify how we integrate the dataset properties as context into the recommendation process as we are looking for improving the prediction and recommendation accuracy.

Chapter 6

Experiments

In this dissertation, we implemented different experiments in order to be able to evaluate our proposed approaches for building an energy-efficient recommender system that considers dataset as a contextual factor.

6.1 Experiments Design

In the set of experiments that we have implemented in relating to our proposed approaches, we used the same elements which include: dataset, web services and the machine as explained in chapter 4. In these proposed approaches, the energy consumption represents the amount of energy required for each single service invocation by the same user while using all the available datasets under study.

In our proposed approaches, we simulate 13 different users from eight different countries to send service requests for invoking the data mining services under study. Each user will send requests to invoke all the existing data mining services using all the available datasets. This kind of setup is close to the real scenario where users may be from many different places.

To simulate different users, we adopted the concept of using Virtual Private Network technology (VPN). For this purpose, we used an open-source tool called Soft-Ether VPN[7] which is easy-to-use multi-protocol VPN software. Accordingly, VPN servers are used from 13 places around the world where requests can be sent to invoke any one of the available web

[7] https://www.softether.org/

services. Our main intention here is to investigate the impact of sending requests from different users to our web server and then examine the impact on the QoS values, more specifically on energy consumption.

We wrote a Java program to automate the process of sending requests from all the users to invoke each data mining service by utilizing our client application and we captured the amount of energy consumption per invocation. The information that we record for each service invocation include: user, web service, dataset ID, dataset properties and associated energy value.

In order to evaluate the prediction and recommendation accuracy, we divide the collected data into two parts: training (70%) and testing (30%). We use the training data to train our recommendation models and then these models are evaluated using the test data.

Our main goal from the implemented experiments is to understand how the properties of the datasets can affect the prediction accuracy of energy consumption as well as the recommendation accuracy. Choosing various services and different datasets helped us gain a broad and clear idea about energy performance. For this purpose, we have implemented two different groups of experiments.

In the first group of experiments, we compare MF model in two situations, the first one when no context is considered in the recommendation engine (standard MF) and the second one when the context (dataset) is considered in the recommendation process. Also, in this set of experiments, we used the complete dataset which has 100 synthetic datasets as we described earlier in Chapter 4, the results of this experiment are presented in section 6.4.1 in this dissertation.

In the second group of experiments, since we want to explore different ways of representing the contextual information and different combinations of dataset properties in our DBCM models, we decided to use only 30 datasets out of the original 100 datasets. The reason for this decision is that many of those datasets are very small in terms of their sizes, although they did vary on different dataset properties; the energy consumption values for those small datasets are low and very close to each other, which give us a skewed data distribution. Keeping all bigger datasets and selecting only some small datasets give us a more balanced dataset, which can reflect the real scenario more closely, the results of this experiment are presented in sections 6.4.2 and 6.4.3 in this dissertation.

From the first group of experiments, we try to find out whether introducing data can improve prediction and recommendation accuracy. From the second group of experiments, we want to identify the model that best fits our research problem. Since we are using synthetic datasets as inputs to the clustering web services, we always have the dataset property values. However, in real applications, sometimes, the dataset properties may be unknown or not accessible, and thus the contextual information is not available. To make our dataset closer to the real scenario, we have intentionally and randomly removed 10% of the contextual information to replace it with "NA".

In relating to the dataset properties we started by using four properties in the first group of experiments, and in the second group of experiments, we keep only three of them because we found that there is a dependency between dataset size and the other properties, which might give some redundancy in our data. Therefore, we decided not to include the file size in the second group of experiments.

We also have conducted various preliminary experiments relating to select the baseline model that we need to use in our evaluations. We started with matrix factorization model as the main baseline model and we showed that in our work presented in [84]. Afterwards, we used the Pearson Correlation Coefficient (PCC) model in two cases, the case of considering the data as context and the case of its standard form where no data (no context is considered) as we presented in [84]. Furthermore, we used the tensor factorization model (TF) as an example of independent contextual modeling and this work is presented in [13]. In our final experiments, we decided to keep only one baseline model as the representative model that we use in the evaluation of each of the proposed approaches; this model is the matrix factorization (MF).

6.2 Evaluation Metrics

In order to evaluate our proposed approaches and comparing them with the baseline model, we used different evaluation metrics as shown in the following sections.

6.2.1 Root Mean Squared Error (RMSE)

This metric is used to measure how close the predictions are to the actual values as shown in equation (6.1).

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(P_i - O_i)^2}{n}} \tag{6.1}$$

Where P_i represents the predicted value, O_i represents the original value and n represents the total number of the observed values.

6.2.2 Precision and Recall

Precision-Recall is a useful measure of success of prediction. In information retrieval, precision is a measure of result relevancy, while recall is a measure of how many truly relevant

results are returned. In other words, precision is defined as the ratio of the number of correctly recommended services overall recommended services; while recall is the ratio of the number of correctly recommended services overall correct services.

High precision relates to a low false positive rate, and high recall relates to a low false negative rate. High scores for precision and recall show that the prediction model is returning accurate results (high precision), as well as returning a majority of all positive results (high recall). A System with high recall but low precision returns many results, but most of its predicted labels are incorrect when compared to the training labels. A system with high precision but low recall is just the opposite, returning very few results, but most of its predicted labels are correct when compared to the training labels. An ideal system with high precision and high recall will return many results, with all results labelled correctly.

Equations (6.2) and (6.3) represent the precision and recall formulas respectively.

$$\text{Precision} = \frac{T_p}{T_p + F_p} \tag{6.2}$$

where T_p represents the number of true positive and F_p represents the number of false positives.

$$\text{Recall} = \frac{T_p}{T_p + F_n} \tag{6.3}$$

where T_p represents the number of true positive and F_n represents the number of false negatives.

6.3 Experiments

6.3.1 Experiment Setup for the Recommendation Approaches

In order to evaluate our proposed approaches, we use the matrix factorization model without context (dataset information) to predict the missing QoS values as the baseline model. After that, we applied the pre-filtering approach (either SBPF or WBPF) to filter the collected data based on dataset similarity then we applied matrix factorization on the filtered data (contextual-filtering approach).

In the experiments to test DBCM models, we would like to compare the contextual modeling approaches with baseline model while using different methods to represent data-related contextual factors. We started by using the standard matrix factorization model (MF) as the baseline model and we use it to compare with our chosen contextual modelling matrix factorization models. Here we have performed three sets of experiments.

In the first set of experiments, we compare DBCM models with the baseline, with all dataset properties included as contextual factors. Once we figure out the best-performing algorithm, in the second set of experiments, we examine the effects of using each one of the dataset properties as a contextual dimension. By doing this, we would like to check whether certain context information is better than the others in terms of enhancing the recommendation accuracy.

We also investigate the impacts of using different combinations of properties to compare with the results we obtained from using a single property. For this purpose, we refer to each one of the dataset properties as P1, P2 and P3, where P1 represents the number of attributes, P2 represents the number of instances and P3 represents the dataset data type. The possible

combinations we consider include the following: a combination of two properties (P1+P2), (P1+P3), and (P2+P3), and a combination of all properties (P1+P2+P3).

In the last set of experiments, we compare two different ways of representing dataset information as contextual factors – using dataset properties as contexts and using the dataset itself as context. We investigate how they differ in terms of recommendation accuracy.

We use CARSKit [77] which is an open-source context-aware recommendation library to test DBCM models. In our preliminary experiments, we have tested different deviation-based models and we decide to choose CAMF-CI (DEV) as a representative for the deviation-based model since it showed the best results in the test. Moreover, we have tested different similarity-based models in order to figure out which one is more suitable for our research problem. We decided to adopt CAMF-LCS (SIM) based on our test results.

6.4 Evaluations of the Proposed Approaches

6.4.1 Similarity-Based Pre-Filtering (SBPF) Approach

Table 6.1 shows RMSE values for energy consumption prediction from the standard MF method and our proposed SBPF approach.

Table 6.1: Root mean squared error obtained from using different prediction methods.

Standard MF	MF with data (SBPF)	Improvement
0.074	0.029	61%

As we can see from Table 6.1, our proposed approach shows a lower level of RMSE, which indicates that the prediction accuracy of this approach is higher than the baseline approach. It is a

61% improvement over the standard MF and this result indicates that considering dataset properties in the recommendation process could largely improve the prediction accuracy.

In order to measure the recommendation accuracy, for each pair of user and service (in the baseline model); and for each set of user, service and dataset (in the proposed approach that considers data); we sorted the list of candidate services based on their predicted energy values in ascending order. Then, we did the same thing based on their actual energy values. Finally, we compared the top service from both lists and count the percentage of successful matches. This percentage value is used as our recommendation accuracy measure. The rationale is that: suppose that we recommend the top service with the lowest predicted energy value to the user, if the result matches with the service with the actual lowest energy value, we consider our recommendation is accurate. Table 6.2 shows the recommendation accuracy of the used approaches.

Table 6.2: Recommendation accuracy from different approaches.

Standard MF	**MF with data (SBPF)**	**Improvement**
31%	63%	32%

As we can see from Table 6.2, our proposed approach (MF with data) shows a higher accuracy for recommending a service to the target user. Furthermore, considering data improve the recommendation accuracy for MF model by 32%.

To conclude this result, we can say that our proposed approach is better than the baseline model in terms of the prediction and recommendation accuracy. This result can confirm the impacts of dataset properties on recommending energy-efficient web services.

6.4.2 Weight-Based Pre-Filtering (WBPF)

Figure 6.1 shows the RMSE values that we obtained from predicting the missing energy values using the baseline model and using our WBPF model. As we can see from the figure, our proposed approach shows lower RMSE value comparing with the standard matrix factorization model.

In relating to recommendation accuracy evaluation, we use the precision and recall metrics. We measure the recommendation accuracy according to the top N results. We have tried N values as 5 and 10, and we found that in most of the cases, the total number of returned results is either less than 5 or around 5. Due to this reason, we have chosen *N* as 5. We will use the same N value in the rest of the experiments. Figure 6.2 illustrates the performance of both WBPF and MF. As we can see from the figure, both precision and recall values of the proposed approach exhibit improvement comparing with the baseline model.

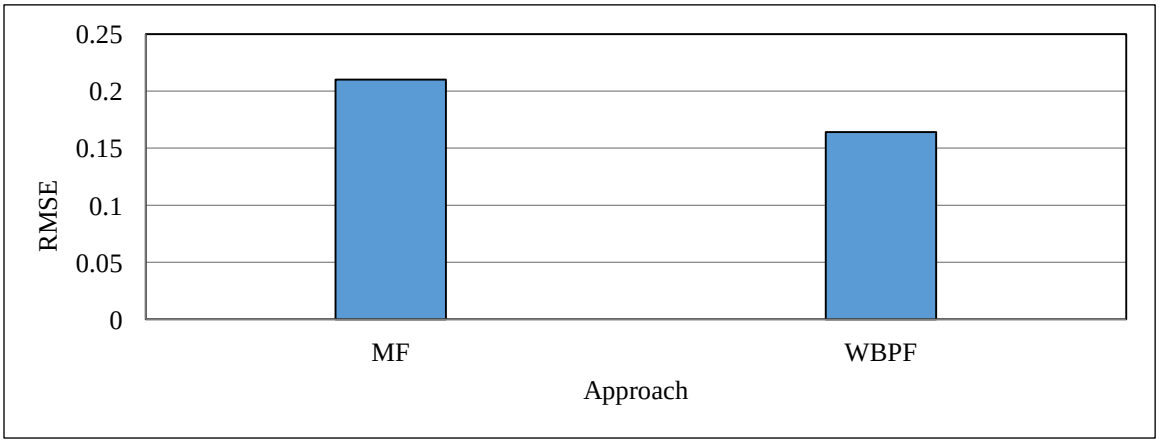

Figure 6.1: Root mean squared error obtained from using MF and WBPF approaches.

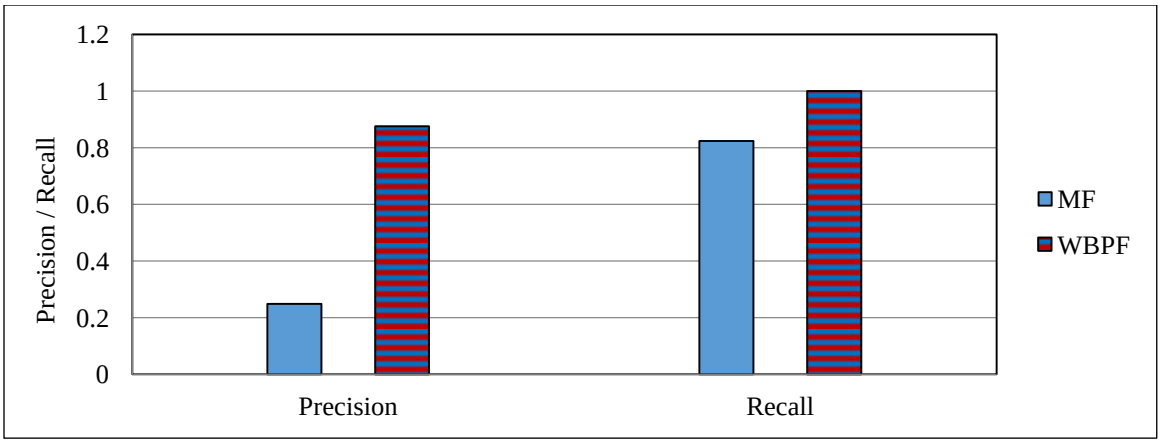

Figure 6.2: Precision and Recall @ Top 5 recommendation obtained from WBPF and MF approaches.

6.4.3 Dataset-Based Contextual Modeling (DBCM) Approach

6.4.3.1 Comparing Different Recommendation Models

In this set of experiments, we compare the results we obtained from incorporating dataset properties as contexts in the recommendation process. Our main goal here is to compare the results we obtained from contextual modeling approaches represented by two models namely: similarity-based (SIM) and deviation-based (DEV), with the baseline model: matrix factorization (MF).

Figure 6.3 shows the results we obtained from this experiment by using all the dataset properties. The results of both evaluation metrics show that both SIM and DEV perform better than the baseline model (MF). In terms of precision, the SIM model performs similar to DEV. On the other hand, MF shows the worse results. For recall metric, DEV is usually better than SIM, while MF shows the worst results.

We have done the same tests on other ways of representing the dataset context and achieved similar results. Contextual modeling approaches always give better recommendation results than the baseline models. Experimental results show that the introduced approach outperforms the

baseline model (MF) by 19.3-22.2% on various evaluation metrics (recall and precision respectively) for measuring recommendation accuracy as illustrated in Figure 6.3. In terms of the processing time of each model, we did not notice a big difference in relating to the time required to recommend the optimal service to the target user among all the proposed approaches.

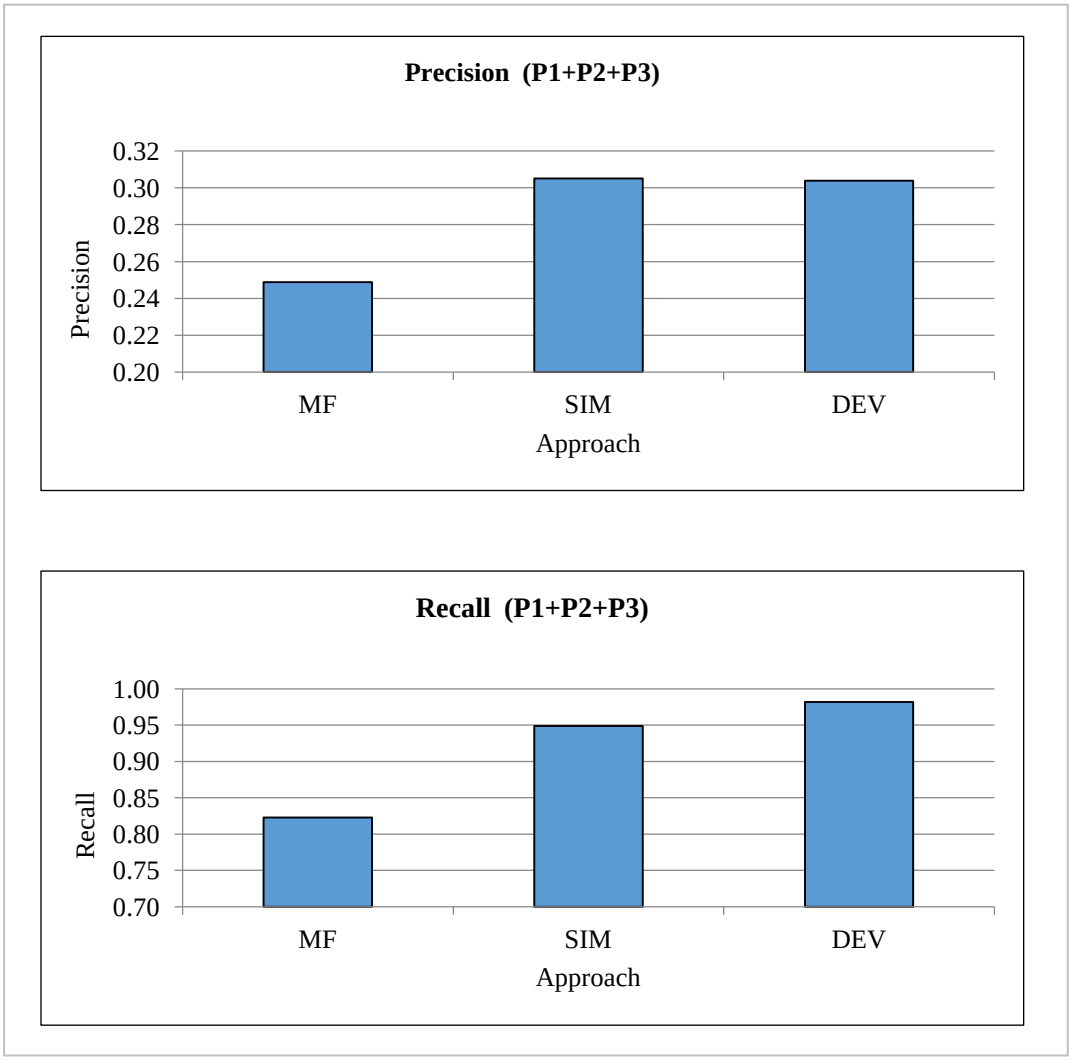

Figure 6.3: Results on precision and recall, using different algorithms and utilizing all dataset properties.

The results show that incorporating dataset properties as contextual factors into the recommendation process can improve recommendation accuracy. Compared with the baseline

model (MF), we found that DEV has achieved an increase of 22.2%, 19.3%, on the precision and recall respectively. When comparing the similarity-based approach with the deviation-based approach, deviation-based approach outperforms the similarity-based approach in most of the cases. This result shows that the deviation approach is more suitable than the similarity approach for our study.

6.4.3.2 Dataset versus Dataset Properties as Contextual Dimensions

In this set of experiments, we investigate the difference between two cases: the case of using dataset properties as contexts and the case of using the dataset itself as context (defined by its ID). For this purpose, we prepare a new dataset that includes user, service and the consumed energy in addition to the ID of the used dataset, as opposed to our original dataset that includes the properties of the dataset (P1, P2, and P3) as contexts.

Figure 6.4 summarizes the results of this experiment. In this experiment, we only show the results we obtained from the deviation-based model because it demonstrates the best performance compared with the similarity-based model (SIM) and the baseline model (MF).

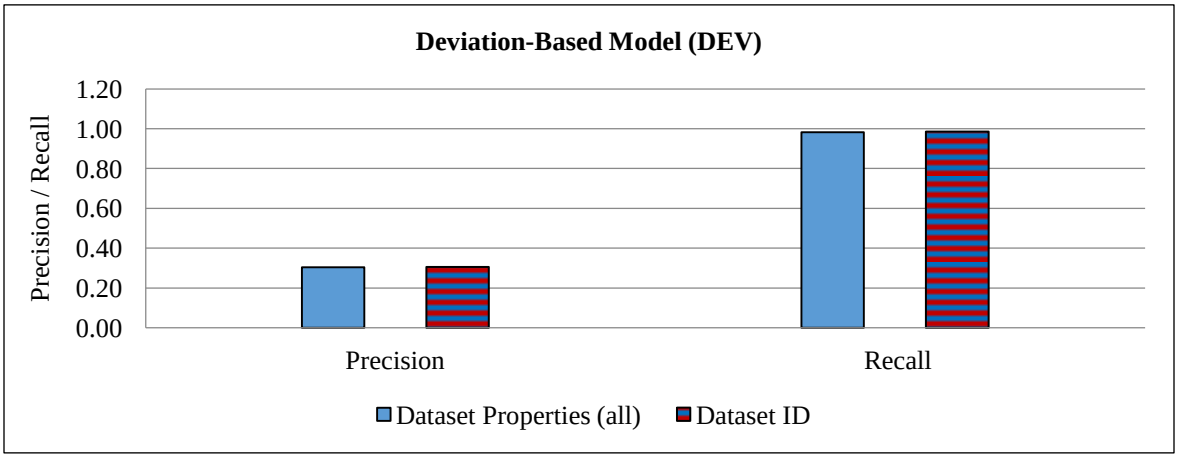

Figure 6.4: Comparison of the evaluation metrics when using dataset properties and dataset as contexts.

As we can see from the results, both precision and recall values are almost the same in the two cases we tested (Dataset ID and dataset properties), which means that using either one can lead to almost the same outcome.

6.4.3.3 Using Different Context Combinations

In this set of experiments, first, we examine the impact of using each dataset property as a separate contextual dimension. Since deviation-based model shows the best result among the contextual modeling approaches and the baseline approaches, here we only show the results from this approach. The results are shown in Figure 6.5. As we can see from the figure, using P2 (number of instances) as a contextual dimension gives a better result compared with the case when using P1 or P3. Both precision and recall metrics show the best results when using P2. When comparing P1 (number of attributes) with P3 (data type), P1 gives better results.

Intuitively, we may conclude that the size of the dataset (which is related to the number of attributes and number of instances) plays an important role in the energy consumption levels when invoking data mining services. This part of our work is presented in [13].

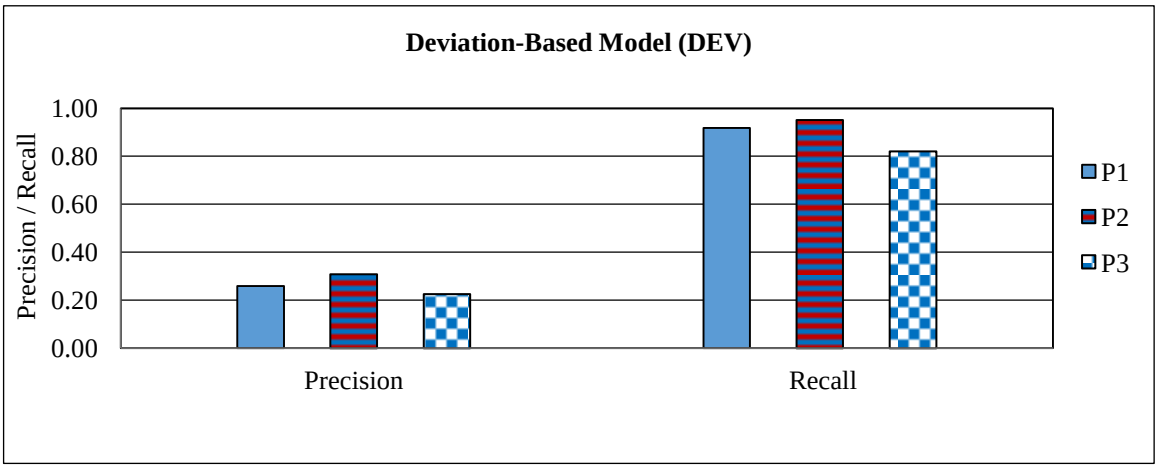

Figure 6.5: Comparison of precision and recall metrics when using different dataset properties.

Table 6.3: Results from using different dataset properties

Properties	Precision	Recall
P1	0.259	0.918
P2	0.309	0.951
P3	0.226	0.821
P1+P2	**0.310**	**0.987**
P1+P3	0.255	0.879
P2+P3	0.306	0.976
P1+P2+P3	0.304	0.982

In the second part of this experiment, we compared various methods of combining the contextual information. For this purpose, we tried combinations of two properties such as (P1+P2), (P1+P3) and (P2+P3). We also tried a combination of all the properties represented by (P1+P2+P3) as shown in Table 6.3.

From Table 6.3 we can see that using P1 and P2 together can give better results than the other combinations; they also show better results compared to the case when only one single property is used as a context. Additionally, using (P1+P2) show better results than (P1+P2+P3), which is the case of considering all the available properties as context. This result is consistent with the previous results in which P2 performs the best, because P2 is part of the best case (P1+P2).

If we check the overall performance, we may say that (P1+P2) gives the best result. It is also interesting to see that by combining all properties together, it doesn't give the best result; this outcome shows the necessity of testing the effectiveness of different contextual factors and adding only the ones (or the combinations) that performs the best into the recommendation model.

6.5 Comparison of the Proposed Approaches

Figure 6.6 shows the RMSE values that we obtained from using both the baseline prediction model (MF) and our proposed context-aware approaches (context-aware solutions). In this comparison, we depend on the second group of experiments because it shows more precise results as explained in section (6.1) and as presented in our paper [13]. As we can see from Figure 6.6, SBPF shows the lowest value of RMSE (0.082) which presents a 61% improvement over the baseline model (MF) that shows 0.21 in relating to RMSE results. RMSE value from WBPF is the second lowest (0.164) which represents a 22% improvement over the baseline model.

From this result, we can conclude that the similarity degree between the target dataset and the existing datasets can enhance the prediction accuracy by 22% - 61% by using WBPF and SBPF respectively. These results prove that our hypothesis is correct in relating to the impact of context similarity on predicting the best fit service to the target user based on the used dataset. The RMSE of DBCM is 0.545, which is worse than the baseline model value.

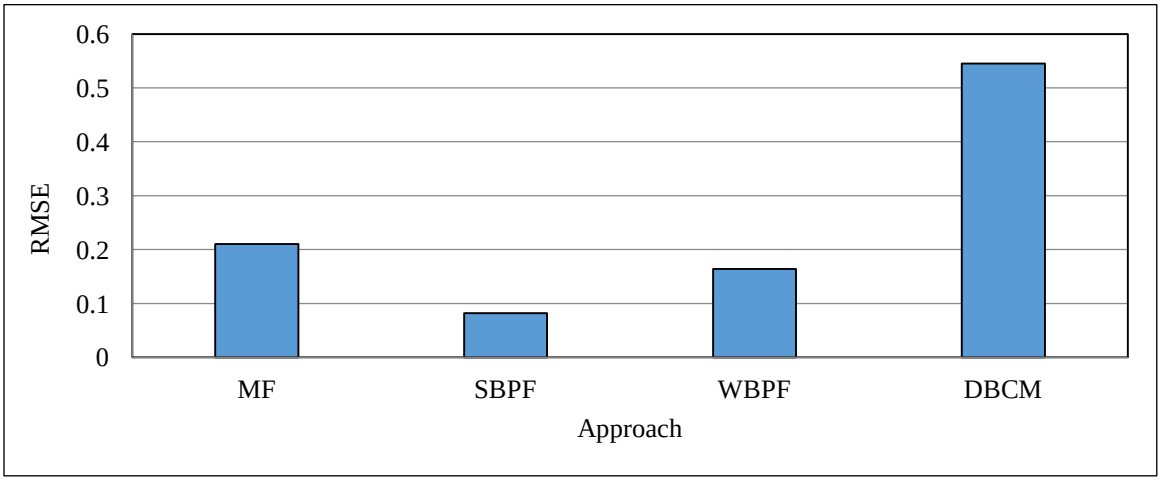

Figure 6.6: Root mean squared error obtained from using the proposed approaches and the baseline model.

In relating to the recommendation accuracy, Figure 6.7 and Figure 6.8 illustrate the precision and recall obtained from the baseline model (MF) and from our proposed approaches respectively. As we can see from Figure 6.7, the baseline model (MF) shows the lowest precision value and all three proposed approaches exhibit higher precision values compared with the baseline model. We can obtain the same outcome from examining Figure 6.8, where the recall values we gained from the proposed approaches are higher than the one from the baseline model. These results show that the inclusion of the dataset properties into the recommendation engine has a clear impact on enhancing the recommendation accuracy of the optimal energy-efficient services.

In relating to the best performing approach from the three proposed ones, we can see that the WBPF approach shows the highest precision compared with SBPF and DBCM. From these results, we can conclude that the weight-based pre-filtering is the approach that best fits our presented research problem. This conclusion is different from what has been reported in the literature [78] that contextual modeling usually produces better recommendation results than the filtering-based approaches. But since the performance of any recommender system is highly dependent on the data, it is not impossible to get a different conclusion than previous research works. In addition to that, in this comparison, we also compare with our new proposed model (WBPF) which represents a new method for implementing the pre-filtering approach.

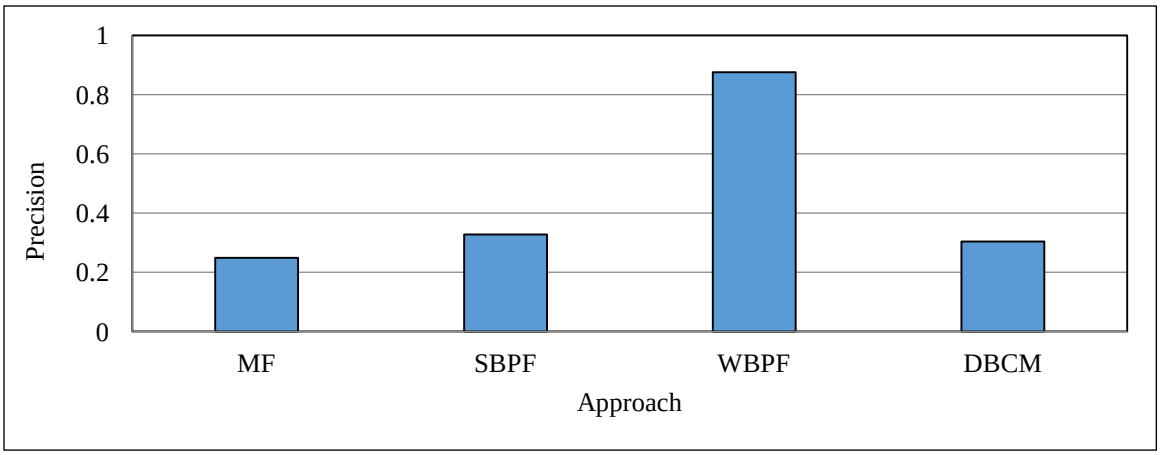

Figure 6.7: Precision measurements obtained as results of using the proposed approaches.

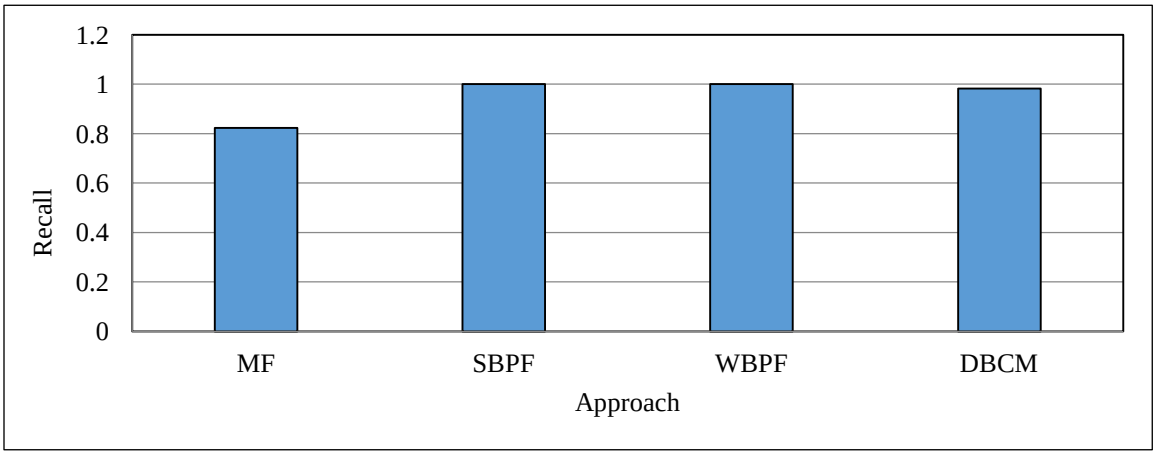

Figure 6.8: Recall measurements obtained as results of using the proposed approaches.

From these experimental results, first, we found that contextual modelling approach (both SIM and DEV) outperforms the standard matrix fractionation. More specifically, the deviation-based approach (DEV) outperforms the similarity-based approach (SIM) in our present context.

Second, we found that although incorporating contextual information has been proven to be able to improve the recommendation accuracy, it is not necessary that having more context dimensions gives better recommendation results in all cases. In this kind of research, we need to choose the contextual information carefully depending on the application and the dataset.

Furthermore, we need to examine various combinations of the available context dimensions as some of them might have higher impacts than the others on the recommendation accuracy. Also, adding unnecessary contextual information can affect the recommendation accuracy in a negative way.

In relating to our study, we found that the number of instances in the dataset is the main factor that we need to consider in order to recommend the energy-efficient service to the user. Further enhancement can be achieved by adding other factors such as the number of attributes and type of datasets.

Lastly, our proposed approaches showed improvement in both the prediction and recommendation accuracy. In terms of the time taken by each approach we notice that the Dataset-Based Contextual Modeling (DBCM) performed better than the other two approaches. Our justification for this outcome is that the pre-filtering process might add extra processing time in both SBPF and WBPF (two pre-filtering approaches), and thus these two models take longer time to run compared to DBCM.

6.6 Discussion

Regarding the best recommendation approach from the three proposed ones, since each one of the proposed approaches showed the best performance when different evaluation metrics were used, we can choose one according to the metric that is more important to the actual problem or to the user. Based on our experimental results, if we really care about how close the predicted energy value is to the actual one, we can use SBPF approach because it provides the best result in prediction accuracy. If the main requirement is to get a high precision value, which means that we are more focusing on the relevancy of the retrieved services, we can use WBPF approach. If

the requirement is to have a high recall value, which means that we want to return as many relevant results as possible, then any of the three approaches would be a good option. Finally, the DBCM can be used in the case of having many contextual dimensions.

6.7 Summary

In this chapter, we have discussed the settings of the experiments in addition to the different types of experiments carried out in this work. Also, we have presented the evaluation metrics which have been used to evaluate the proposed approaches. The evaluation of each one of the proposed approaches was then discussed and then compared with the baseline models to examine the changes in both prediction accuracy and recommendation accuracy as a result of integrating dataset properties as contextual factors. In the end, all the proposed approaches were compared with each other to determine which one is best suited to the presented research problem.

Chapter 7
Conclusions and Future Works

7.1 Conclusions

In this dissertation, we have proposed context-aware approaches for recommending energy-efficient data mining services (clustering services). The proposed approaches are based on the findings of our study on the behaviour of energy consumption and latency resulting from using a number of clustering services. The study shows that the properties of the dataset can affect the energy consumption of web services (and other QoS values such as latency). Based on this finding, we propose three approaches for predicting energy consumption values of data mining services and then rank these services based on their predicted values.

In our proposed approaches, we have introduced data as a contextual factor that can be integrated into the web service recommender system. We have implemented it using a context-aware matrix factorization model. In our proposed recommender system, energy consumption is the main QoS attribute that is considered in the recommendation process.

Additionally, in this dissertation, we investigate the impact of using various methods of representing and integrating the contextual information into the recommendation engine. We further investigate the impacts of using different combinations of the available contextual information, represented by the dataset properties in this work, and how this action might affect the accuracy of recommending the optimal service.

In the conducted experiments, we examined both prediction accuracy and recommendation accuracy, by comparing our model with existing baseline models. Our experiments have shown that both prediction and recommendation results are better than the results obtained from the baseline model. Among the proposed approaches, the weight-based pre-filtering approach achieved the best performance.

In this dissertation we propose to use energy consumption as the main QoS attribute in the proposed recommender system. Other QoS properties can be considered too, such as response time, latency and availability. In relating to the best recommendation approach from the proposed ones, we found that each one of the proposed approaches can be used in a specific situation and different approach can be chosen based on the actual requirements. For example, in situation when the prediction is the main request, one can use SBPF approach. If the relevancy is the main concern, WBPF approach would be the best choice. In the case of having many contextual dimensions, the DBCM approach would be the best choice.

Our contributions from this work have important benefits to service providers who usually look to cut the operational cost of their services, which is in turn affected by the amount of energy consumption associated with these services.

7.2 Contributions

The main contributions of this dissertation can be described as follows:

- Proposed energy-efficiency as one of the important QoS properties that can be used in selecting and recommending optimal data mining web services with the least level of energy consumption. Since most of the existing QoS-based service selection algorithms have not considered energy consumption in the selection process, we have conducted an

experimental study on how using different datasets that have various attributes can affect the energy consumption and latency of data mining services. Our study shows that the energy consumption level of data mining services can be affected by the used dataset attributes such as the number of attributes, number of instances, and data type. Furthermore, we reveal that different implementations or providers (in our case we use Weka, R and Apache) of services also have an impact on energy consumption.

- Introduced dataset properties as contextual information that can be integrated into three proposed recommendation approaches. The accuracy of both energy value prediction as well as recommending the energy-efficient services have been improved as a result of using our proposed approaches. To the best of our knowledge, this is the first work to address the importance of data (dataset properties) in recommending energy-efficient data mining web services.
- Built a framework designed for collecting real QoS data (energy consumption) of data mining services (clustering services) by using a set of synthetic datasets varying on their properties and being invoked by different users. Although the datasets and users are simulated, the data mining services and their QoS data are real; the collected data is then used to validate the proposed approaches.

Table 7.1 highlights the research contributions from both the theoretical and technical perspectives.

Table 7.1: Research contributions from both the theoretical and technical perspectives.

	Theoretical Contributions	Technical Contributions
Contribution 1	Identify energy consumption as an important QoS attribute to consider when recommending services.	-Design and implement a framework for collecting the related data. -Create a QoS dataset (energy and latency) with contextual information (properties of the dataset for the corresponding invocation), which can potentially be used by other researchers on study related to context-aware service recommender systems or data mining recommender systems.
Contribution 2	Introduce dataset properties as contextual information.	-Through experimental study, find the relationships between dataset properties and QoS properties (energy and latency). -Through experimental study, find the relationships between dataset properties and recommendation accuracy.
Contribution 3	Propose Similarity-Based Pre-Filtering approach for implementing our context-aware recommender system.	-Use a new implementation for the pre-filtering approach based on the used dataset properties as contexts which are used in the filtering process. This includes measuring the similarity degree between the datasets and then implementing the filtering process accordingly. -Apply the matrix factorization model based on the filtered data.
	Propose Weight-Based Pre-Filtering for implementing our context-aware recommender system.	-Use a new method for implementing the pre-filtering technique. The proposed method focuses on using weighted factors measured based on the similarity degree between the target dataset and all the remaining datasets in the dataset repository. -Use a new implementation for calculating the similarity degree between the datasets, calculating the new rating values based on the similarity values, implementing the filtering process. -Apply the matrix factorization model based on the filtered data.
	Propose Dataset-Based Contextual Modeling: for implementing our context-aware recommender system.	-Use different methods for representing the contextual information, with dataset properties, or with dataset IDs. -Introduce the idea of using different combinations of the available contextual information rather than using individual contexts or using all the contexts, in order to disclose the ones with the high recommendation accuracy.

7.3 Limitations of Our Work

Part of the information that we wanted to collect in this dissertation is the energy consumption levels of the invoked data mining services. Since it is not accessible for us to measure the energy consumption of the real web services published online, we decided to integrate our proposed energy measurement framework with the testbed system presented in [54]. Based on this decision, we used some of the data mining web services implemented in the mentioned system. Due to the hardware constraints when running the experiment, the maximum size of the dataset we can successfully run all the data mining services within reasonable time is 40 MB, as shown in Table 4.1. This is considered as a limitation of our current work. Furthermore, our implemented experiments are based on one dataset where we have saved the user, service, datasets, amount of energy consumption and other QoS values. To make more generalized conclusion, we should have run our experiment on more datasets. Additionally, we tested our proposed approach on one server and we did not test it in a distributed cloud environment.

7.4 Future Work

In this dissertation, we only consider clustering data mining algorithms, and as for the QoS properties, we only consider energy consumption. In the future, we are interested in expanding this research work to use different types of data mining services such as classification and association rule mining. Moreover, we would like to use more QoS properties of web services such as response time, reliability and availability, and then study their relationship with web service's energy consumption.

We are also interested in examining the effects of increasing the amount of historical QoS data and how it might affect both prediction and recommendation accuracy. Therefore, we are planning to test our proposed approaches on a bigger-scaled dataset, with more input datasets, more dataset properties, and more varying property values. We are also planning to investigate other possible options to add dataset properties into the existing recommender system seeking for more improvement in recommending energy-efficient data mining web services. In addition, we are interested in examining the relationship between other factors such as CPU and RAM utilization per individual web service and the amount of energy consumption. Furthermore, more work needs to be done in order to consider the case of having a big dataset hosted in multiple computers.

Finally, we plan to consider the situation of having parallel users using the proposed system at the same time and to formalize our discussed problem as an optimization problem with objective functions defined over the used QoS values.

Bibliography

[1] A. V. Miranskyy, Z. Al-zanbouri, D. Godwin, and A. B. Bener, "Database engines: Evolution of greenness," *Journal of Software: Evolution and Process*, vol. 30, no. 4, p. e1915, 2018.

[2] I.A.T. Hashem, I. Yaqoob, N.B. Anuar, S. Mokhtar, A. Gani, and S.U. Khan, "The rise of "big data" on cloud computing: Review and open research issues", *Information Systems*, 47: 98115, (pp.98-115) ,2015.

[3] Y. Jiang, J. Liu, M. Tang and X. Liu. "An effective web service recommendation method based on personalized collaborative filtering". In *Web Services (ICWS), IEEE International Conference on* (pp. 211-218). IEEE, July 2011.

[4] S.Y. Hwang, C.C. Hsu, and C.H. Lee, "Service selection for web services with probabilistic QoS", *IEEE Transactions on Services Computing*, 8(3): 467-480, 2015.

[5] Z.Al-Zanbouri, and C. Ding, "Selecting Green Data Mining Services". In *Services Computing (SCC), 2017 IEEE International Conference* on (pp. 100-107). IEEE, 2017, June.

[6] P Singh and A Surya, "Performance analysis of clustering algorithms in data mining in weka", *International Journal of Advances in Engineering & Technology*, 7(6): 1866-1873, 2015.

[7] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver, "Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering". In *Proceedings of the fourth ACM conference on Recommender systems* (pp. 79-86). ACM, September 2010.

[8] L. Baltrunas and F. Ricci, "Context-based splitting of item ratings in collaborative filtering", In *Proceedings of the third ACM conference on Recommender systems* (pp. 245-248). ACM, October 2009.

[9] Y. Shi, M. Larson and A. Hanjalic. "Mining mood-specific movie similarity with matrix factorization for context-aware recommendation", In *Proceedings of the workshop on context-aware movie recommendation* (pp. 34-40). ACM, Barcelona, Spain, September 2010.

[10] L. Baltrunas, B. Ludwig, S. Peer and F. Ricci, F, "Context relevance assessment and exploitation in mobile recommender systems". *Personal and Ubiquitous Computing*, 16(5), 507-526, 2012.

[11] J. Liu, M. Tang, Z. Zheng, X. F. Liu and S. Lyu, "Location-aware and personalized collaborative filtering for web service recommendation". *IEEE Transactions on Services Computing*, *9*(5), 686-699, 2016.

[12] Y. Zheng, B. Mobasher and R. Burke, "Similarity-based context-aware recommendation". In *International Conference on Web Information Systems Engineering* (pp. 431-447). Springer, Cham, 2015, November.

[13] Z. Al-Zanbouri and C. Ding, “Recommending Energy-Efficient Data Mining Services with Data as Contextual Factors,” in *2019 IEEE International Conference on Services Computing (SCC)*, 2019, pp. 14–18.

[14] P. Bartalos, Y. Wei, M. B. Blake, H. Damgacioglu, I. Saleh, and N. Celik, “Modeling energy-aware web services and application,” *Journal of Network and Computer Applications*, vol. 67, pp. 86–98, May 2016.

[15] G. Adomavicius and A.Tuzhilin, “Context-aware recommender systems”. In *Recommender systems handbook* (pp. 191-226). Springer, Boston, MA, 2015.

[16] G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin, “Incorporating contextual information in recommender systems using a multidimensional approach,” *ACM Transactions on Information Systems (TOIS)*, vol. 23, no. 1, pp. 103–145, 2005.

[17] C.J. Lin, “Projected Gradient Methods for Nonnegative Matrix Factorization,” *Neural Computation*, vol. 19, no. 10, Oct. 2007, pp. 2756–2779.

[18] L. Baltrunas, B. Ludwig, and F. Ricci, “Matrix factorization techniques for context aware recommendation,” in *Proceedings of the fifth ACM conference on Recommender systems*, 2011, pp. 301–304.

[19] Y. Zheng, B. Mobasher, and R. Burke, “Incorporating Context Correlation into Context-aware Matrix Factorization,” In *CPCR+ ITWP@ IJCAI.*,2015, *p. 7.*

[20] N. Jain, C. Ding, and X. Liu, “Data-Dependent QoS-based service selection”, in *Proceedings of International Conference on Service-Oriented Computing*, pp. 617–625, 2016.

[21] W. Ahmed, Y. Wu, and W. Zheng, “Response time based optimal web service selection”, *IEEE Transactions on Parallel and Distributed Systems*, 26(2): 551–561, Feb. 2015.

[22] S.Y. Hwang, H. Wang, J. Srivastava, and R. A. Paul, “A probabilistic QoS model and computation framework for web services-based workflows”, in *Proceedings of the 23rd International Conference on Conceptual Modeling*, pp. 596–609, 2004.

[23] S. Deng, H. Wu, D. Hu, and J. L. Zhao, “Service selection for composition with QoS correlations”, *IEEE Transactions on Services Computing*, 9(2): 291–303, Mar. 2016.

[24] X. Liu, W. Shi, A. Kale, C. Ding, and Q. Yu, “Statistical Learning of Domain-Specific Quality-of-Service Features from User Reviews,” *ACM Trans. Internet Technol.*, vol. 17, no. 2, pp. 22:1–22:24, May 2017.

[25] Z. Zheng, Y. Zhang, and M. R. Lyu, “Investigating QoS of real-world web services,” *IEEE transactions on services computing*, vol. 7, no. 1, pp. 32–39, 2012

[26] P. J. Stockreisser, J. Shao, W. A. Gray, and N. J. Fiddian, “Supporting QoS monitoring in virtual organizations,” in *International Conference on Service-Oriented Computing*, 2006, pp. 447–452.

[27] F. Rosenberg, C. Platzer, and S. Dustdar, “Bootstrapping performance and dependability attributes of web services,” in *2006 IEEE International Conference on Web Services (ICWS’06)*, 2006, pp. 205–212.

[28] A. Amin, A. Colman, and L. Grunske, “An Approach to Forecasting QoS Attributes of Web Services Based on ARIMA and GARCH Models,” in 2012 IEEE 19th International Conference on Web Services, 2012, pp. 74–81.

[29] H. Zheng, J. Yang, W. Zhao, and A. Bouguettaya, “QoS analysis for web service compositions based on probabilistic QoS,” in International Conference on Service-Oriented Computing, 2011, pp. 47–61.

[30] A. Kattepur, N. Georgantas, and V. Issarny, “QoS composition and analysis in reconfigurable web services choreographies,” in 2013 IEEE 20th International Conference on Web Services, 2013, pp. 235–242.

[31] S. Rangarajan, “Qos-Based Web Service Discovery And Selection Using Machine Learning,” ICST Transactions on Scalable Information Systems, vol. 5, no. 17, p. 154809, May 2018.

[32] H. Wang, C. Yu, L. Wang, and Q. Yu, “Effective Big Data-Space Service Selection over Trust and Heterogeneous QoS Preferences,” *IEEE Transactions on Services Computing*, vol. 11, no. 4, pp. 644–657, Jul. 2018.

[33] Kirit J. Modi, Sanjay Garg, (2019) "A QoS-based approach for cloud-service matchmaking, selection and composition using the Semantic Web", Journal of Systems and Information Technology, Vol. 21 Issue: 1, pp.63-89.

[34] Y. Zhou, Z. Tang, L. Qi, X. Zhang, W. Dou, and S. Wan, "Intelligent Service Recommendation for Cold-Start Problems in Edge Computing," *IEEE Access*, vol. 7, pp. 46637–46645, 2019.

[35] S. K. Gudla, J. Bose, and K. R. Sane, "Enhanced Service Recommender and Ranking System Using Browsing Patterns of Users," in *2019 16th IEEE Annual Consumer Communications Networking Conference (CCNC)*, 2019, pp. 1–2.

[36] G.Tian, Q. Wang, J. Wang, K. He, W. Zhao, P. Gao and Y. Peng "Leveraging contextual information for cold-start Web service recommendation: Leveraging contextual information for cold-start web service recommendation," *Concurrency and Computation: Practice and Experience*, p. e5195, Feb. 2019.

[37] B. Cao, B. Li, J. Liu, M. Tang, Y. Liu, and Y. Li, "Mobile Service Recommendation via Combining Enhanced Hierarchical Dirichlet Process and Factorization Machines," *Mobile Information Systems*, vol. 2019, 2019.

[38] Y. Yin, L. Chen, Y. Xu, J. Wan, H. Zhang, and Z. Mai, "QoS prediction for service recommendation with deep feature learning in edge computing environment," *Mobile Networks and Applications*, pp. 1–11, 2019.

[39] J. Lu, G. Liu, K. Wu, and W. Qin, "Location-Aware Web Service Composition Based on the Mixture Rank of Web Services and Web Service Requests," *Complexity*, vol. 2019, 2019.

[40] J. Huang and C. Lin, "Agent-based green web service selection and dynamic speed scaling", in *Proceedings of the 20th IEEE International Conference on Web Services*, pp. 91–98, 2013.

[41] E. Gelenbe, and Y. Caseau, "The impact of information technology on energy consumption and carbon emissions", *Ubiquity, 2015*(June): 1-15, 2015.

[42] C. Comito and D. Talia, "Evaluating and predicting energy consumption of data mining algorithms on mobile devices", in *Proceedings of the IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1–8, 2015.

[43] Y. Zhu, M. Halpern, and V. J. Reddi, "Event-based scheduling for energy-efficient QoS (eQoS) in mobile web applications", in *Proceedings of the 21st IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pp. 137–149, 2015.

[44] P. Bartalos, Y. Wei, M. B. Blake, H. Damgacioglu, I. Saleh, and N. Celik, "Modeling energy-aware web services and application", *Journal of Network and Computer Applications*, 67: 86–98, May 2016.

[45] B. Zhao, W. Hu, Q. Zheng, and G. Cao, "Energy-Aware web browsing on smartphones", *IEEE Transactions on Parallel and Distributed Systems*, 26(3): 761–774, Mar.2015.

[46] J. Liu, J. Jiang, X. Cui, W. Yang, and X. Liu, "Power consumption prediction of web services for energy-efficient service selection", *Personal and Ubiquitous Computing*, 19(7): 1063–1073, Oct. 2015.

[47] Y. Ngoko, A. Goldman, and D. Milojicic, "Service selection in web service compositions optimizing energy consumption and service response time", *Journal of Internet Services and Applications*, vol. 4, no. 1, p. 19, Nov. 2013.

[48] W. Lang, S. Harizopoulos, J. M. Patel, M. A. Shah, and D. Tsirogiannis, "Towards Energy-Efficient Database Cluster Design", *Proceedings of the VLDB Endowment 5(11)*: 1684-1695, 2012.

[49] A. Roukh, L. Bellatreche, and C. Ordonez, "EnerQuery: Energy-Aware Query Processing", in *Proceedings of the 25th ACM International on Conference on Information and Knowledge*, pp. 2465-2468, 2016.

[50] P. Neophytou, J. Szwedko, M. A. Sharaf, P. K. Chrysanthis, and A. Labrinidis, "Optimizing the Energy Consumption of Continuous Query Processing with Mobile Clients", *Mobile Data Management (1):* 98-103, 2011.

[51] L. Brahimi, Y. Ouhammou, L. Bellatreche, and A. Ouared, "More transparency in testing results: Towards an open collective knowledge base", in *Proceedings of the IEEE Tenth International Conference on Research Challenges in Information Science (RCIS)*, pp. 1-6, 2016.

[52] Ma, X., Lin, C., Zhang, H., & Liu, J., "Energy-aware computation offloading of IoT sensors in cloudlet-based mobile edge computing". Sensors, 18(6), 1945, 2018.

[53] O. Alsaryrah, I. Mashal, and T.-Y. Chung, "Energy-aware services composition for Internet of Things," in *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*, Singapore, 2018, pp. 604–608.

[54] M. S. Rahman, C. Ding, X. Liu, and C. H. Chi, "A testbed for collecting qos data of cloud-based analytic services", in *Proceedings of the 9th IEEE International Conference on Cloud Computing (CLOUD)*, pp. 236–243, 2015.

[55] S. A. Mohamed, T. H. A. Soliman and A. A. Sewisy, "A context-aware recommender system for personalized places in mobile applications". *Int. J. Adv. Comput. Sci. Appl*, *7*(3), 442-448, 2016.

[56] U.Panniello and M. Gorgoglione, "A contextual modeling approach to context-aware recommender systems". In *Proceedings of the 3rd Workshop on Context-Aware Recommender Systems*, vol. 1, No. 52, p. 2, October 2011.

[57] Y. Zheng and A. A. Jose, "Context-aware recommendations via sequential predictions," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing - SAC '19*, Limassol, Cyprus, 2019, pp. 2525–2528.

[58] J. Xie, F. Zhu, M. Huang, N. Xiong, S. Huang, and W. Xiong, "Unsupervised Learning of Paragraph Embeddings for Context-Aware Recommendation," *IEEE Access*, vol. 7, pp. 43100–43109, 2019.

[59] Y. Zheng, B. Mobasher, and R. Burke, “CSLIM: Contextual SLIM recommendation algorithms,” in *Proceedings of the 8th ACM Conference on Recommender Systems*, 2014, pp. 301–304.

[60] M. Iqbal, M.A. Ghazanfar, A. Sattar, M. Maqsood, S.Khan, I. Mehmood and S. W. Baik, “Kernel Context Recommender System (KCR): A Scalable Context-Aware Recommender System Algorithm,” *IEEE Access*, vol. 7, pp. 24719–24737, 2019.

[61] M. Gan, Y. Ma, and K. Xiao, “CDMF: A Deep Learning Model based on Convolutional and Dense-layer Matrix Factorization for Context-Aware Recommendation,” p. 8.

[62] V. S. Dixit and P. Jain, “Weighted Percentile-Based Context-Aware Recommender System,” in *Applications of Artificial Intelligence Techniques in Engineering*, 2019, pp. 377–388.

[63] A. Tripathi, T. S. Ashwin, and R. M. R. Guddeti, “EmoWare: A Context-Aware Framework for Personalized Video Recommendation Using Affective Video Sequences,” *IEEE Access*, vol. 7, pp. 51185–51200, 2019.

[64] X. Li, Z. Wang, S. Gao, R. Hu, Q. Zhu, and L. Wang, “An Intelligent Context-Aware Management Framework for Cold Chain Logistics Distribution,” *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–14, 2019.

[65] G. Xu, L. He, and M. Hu, “Document Context-Aware Social Recommendation Method,” in *2019 International Conference on Computing, Networking and Communications (ICNC)*, 2019, pp. 787–791.

[66]Y. Koren, R. Bell, and C. Volinsky, “Matrix factorization techniques for recommender systems,” *Computer*, no. 8, 2009, pp. 30–37.

[67] Q. Chen, X. Song, Z. Fan, T. Xia, H. Yamada, and R. Shibasaki, “A Context-Aware Nonnegative Matrix Factorization Framework for Traffic Accident Risk Estimation via Heterogeneous Data,” in *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, 2018, pp. 346–351.

[68]Y. Fang and Y. Guo, “A context-aware matrix factorization recommender algorithm,” in *2013 IEEE 4th International Conference on Software Engineering and Service Science*, 2013, pp. 914–918.

[69] Y. Zheng, B. Mobasher and R. Burke. “Correlation-Based Context-aware Matrix Factorization”. In *Proceedings of School of Computing Research Symposium (SOCRS)*, DePaul University, Chicago, USA, May 2015.

[70] R. Karim, C. Ding, A. Miri, and M. S. Rahman, “End-to-End QoS Prediction Model of Vertically Composed Cloud Services via Tensor Factorization”, in 2015 *International Conference on Cloud and Autonomic Computing*, 2015, pp. 158–168.

[71] M. Zhang, X. Liu, R. Zhang, and H. Sun, “A Web Service Recommendation Approach Based on QoS Prediction Using Fuzzy Clustering”, in 2012 *IEEE Ninth International Conference on Services Computing*, 2012, pp. 138–145.

[72] J. Zhu, P. He, Z. Zheng, and M. R. Lyu, “A Privacy-Preserving QoS Prediction Framework for Web Service Recommendation”, In *Web Services (ICWS), 2015 IEEE International Conference* Jun. 2015, pp. 241–248.

[73] Y. Chen and Z. Jiang, “Dynamically predicting the quality of service: batch, online, and hybrid algorithms,” *Journal of Electrical and Computer Engineering*, vol. 2017, 2017.

[74] Wattsup, https://www.wattsupmeters.com/, last accessed at Feb 3,2017.

[75] Joule meter, https://www.microsoft.com/en-us/research/project/joulemeter-computational-energy-measurement-and-optimization/, last accessed at Feb 3, 2017.

[76] R. Karim, C. Ding, and C.-H. Chi, “An enhanced PROMETHEE model for QoS-based web service selection,” in *2011 IEEE International Conference on Services Computing*, 2011, pp. 536–543.

[77] Y. Zheng, B. Mobasher, and R. Burke, “Carskit: A java-based context-aware recommendation engine,” in *Data Mining Workshop (ICDMW), 2015 IEEE International Conference on*, 2015, pp. 1668–1671.

[78] Y. Zheng, “Interpreting Contextual Effects By Contextual Modeling In Recommender Systems,” *arXiv preprint arXiv:1710.08516*, 2017.

[79] A. Karahoca, *Advances in Data Mining Knowledge Discovery and Applications*. BoD – Books on Demand, 2012.

[80] S. Wang, Y. Ma, B. Cheng, F. Yang, and R. N. Chang, “Multi-dimensional QoS prediction for service recommendations,” *IEEE Transactions on Services Computing*, vol. 12, no. 1, 2016, pp. 47–57.

[81] G. Adomavicius and A. Tuzhilin, “Context-aware recommender systems,” in *Recommender systems handbook*, Springer, 2011, pp. 217–253.

[82] Y. Zheng, R. Burke, and B. Mobasher, “Differential context modeling in collaborative filtering,” in *Proceedings of School of Computing Research Symposium. DePaul University, Chicago IL, USA*, 2013.

[83] Y. Zheng, R. Burke, and B. Mobasher, “Recommendation with differential context weighting,” in *International Conference on User Modeling, Adaptation, and Personalization*, 2013, pp. 152–164.

[84] Z. Al-Zanbouri and C. Ding, “Data-Aware Web Service Recommender System for Energy-Efficient Data Mining Services,” in *2018 IEEE 11th Conference on Service-Oriented Computing and Applications (SOCA)*, 2018, pp. 57–64.