DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

MODELLING USER HABITS AND PROVIDING RECOMMENDATIONS BASED ON HYBRID TELEVISION STANDARDS USING ARTIFICIAL NEURAL NETWORKS TOGETHER WITH GENETIC ALGORITHMS

by
İhsan TOPALLI

January, 2017
İZMİR

MODELLING USER HABITS AND PROVIDING RECOMMENDATIONS BASED ON HYBRID TELEVISION STANDARDS USING ARTIFICIAL NEURAL NETWORKS TOGETHER WITH GENETIC ALGORITHMS

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in Electrical and Electronics Engineering

by
İhsan TOPALLI

January, 2017
İZMİR

Ph.D. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"MODELLING USER HABITS AND PROVIDING RECOMMENDATIONS BASED ON HYBRID TELEVISION STANDARDS USING ARTIFICIAL NEURAL NETWORKS TOGETHER WITH GENETIC ALGORITHMS"** completed by **İHSAN TOPALLI** under supervision of **ASSOC. PROF. DR. SELÇUK KILINÇ** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy.

Assoc. Prof. Dr. Selçuk KILINÇ

Supervisor

Assoc. Prof. Dr. Olcay AKAY

Thesis Committee Member

Assoc. Prof. Dr. Adil ALPKOÇAK

Thesis Committee Member

Prof. Dr. Cüneyt GÜZELİŞ

Examining Committee Member

Prof. Dr. Serdar ÖZOĞUZ

Examining Committee Member

Prof.Dr. Emine İlknur CÖCEN

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGMENTS

First of all, I would like to thank my advisor Assoc. Prof. Dr. Selçuk KILINÇ for his continuous guidance. He has always encouraged, motivated, and trusted me throughout this research.

I wish to express my gratitude to the examining committee members Assoc. Prof. Dr. Olcay AKAY and Assoc. Prof. Dr. Adil ALPKOÇAK for showing close attention to my work and for making valuable comments and suggestions since the beginning of the thesis.

I am thankful to Melek KUMLUCA, Songül GÜNDÜZ, Beyhan KARATEPE, Meryem ÇAĞLAYAN, Emine KONAŞOĞLU, Nermin GÜNDÜZ, and Cumhur GÜNDÜZ who provided the valuable data for my work by recording every television programs they have watched for more than a month. There are also 248 people who answered the questionnaire about their genre preferences. Even though I don’t know most of them personally, I am thankful to all of them as well.

Last but not the least, I would like to thank my spouse Ayça KUMLUCA TOPALLI. This thesis would not be complete without her continuous support and motivation.

İhsan TOPALLI

MODELLING USER HABITS AND PROVIDING RECOMMENDATIONS BASED ON HYBRID TELEVISION STANDARDS USING ARTIFICIAL NEURAL NETWORKS TOGETHER WITH GENETIC ALGORITHMS

ABSTRACT

In this thesis, a novel method to make smart recommendations is proposed utilizing artificial intelligence and the latest technologies developed for the television area. For this purpose, genetic algorithms (GAs), artificial neural networks (ANNs), and Hybrid Broadcast Broadband Television (HbbTV) are combined to get the users' television viewing habits and to create profiles. Then, television programs are recommended to the users based on that profiling.

An HbbTV application is developed to collect real television watching data from the users. Then, these data are employed to cluster users. The number of clusters is found by "Controlled Clustering with Genetic Algorithms (CCGA)", a method proposed in this thesis. For each cluster formed by CCGA, a separate ANN is designed to learn the viewing habits of the users of the corresponding cluster. The weight matrices are initialized also by GA. The constructed model is then used to provide recommendations to the users again using the same HbbTV application.

The novelty of this work lies in several areas. Since it is mainly a broadcaster side method and HbbTV is a widely accepted public standard, this proposal is device-agnostic and can reach many people using different device brands, which is the main motivation of this study.

Clustering the users enhances the learning part, which is based on the ANN. Instead of assigning all users to the same ANN, clustering is introduced by utilizing preferred genre information obtained explicitly. GA is one of the clustering alternatives but it proves itself as a better approach when compared to the well-known K-means clustering algorithm.

Introducing a penalizing transformation in clustering with GA keeps the ANN network size under control which is an important parameter in terms of processing cost and time. Similarly, starting the ANN learning with pre-processed weights rather than random values improves the performance.

Keywords: Artificial neural networks, automatic detection of number of clusters, CCGA, clustering, data collection, digital receivers, genetic algorithms, HbbTV, K-means algorithm, multilayer perceptron, program recommendation, recommendation engine, user profiling

YAPAY SİNİR AĞLARI İLE BİRLİKTE GENETİK ALGORİTMALAR KULLANILARAK İZLEYİCİ ALIŞKANLIKLARININ KARMA TELEVİZYON STANDARTLARI TABANLI MODELLENMESİ VE ÖNERİ OLUŞTURULMASI

ÖZ

Bu tezde, yapay zekâ ve televizyon alanında geliştirilen son teknolojiler ile akıllı öneriler yapan yeni bir yöntem sunulmuştur. Bunu yapabilmek için, genetik algoritma (GA), yapay sinir ağı (YSA) ve Karma Yayın Genişbant Televizyon (KygTV) birleştirilerek kullanıcının televizyon izleme alışkanlıkları elde edilmiş ve profiller yaratılmıştır. Sonrasında bu profilleme baz alınarak kullanıcıya TV programları önerilmiştir.

Bir KygTV uygulaması geliştirilerek gerçek televizyon izleme verisi toplanmıştır. Bu veri daha sonra kullanıcıları kümelemek için kullanılmıştır. Küme sayısı, bu tezde önerilen bir yöntem olan “GA ile Kontrollü Kümeleme (GAKK)” ile bulunmuştur. GAKK ile oluşturulan her küme için, ilgili kümenin kullanıcılarının izleme alışkanlıklarını öğrenmek üzere ayrı bir YSA tasarlanmıştır. Ağırlık dizeylerinin ilk değerleri de GA ile belirlenmiştir. Oluşturulan model sonrasında kullanıcılara aynı KygTV uygulaması ile öneriler sunmak için kullanılmıştır.

Bu çalışmanın yeniliği birkaç alanda yer almaktadır. Bu, asıl olarak yayıncı tarafı ile ilgili bir yöntem olduğu ve KygTV yaygınca kabul edilmiş açık bir tüketici elektroniği standardı olduğu için önerilen bu yöntem alıcıdan bağımsızdır ve farklı alıcı markaları kullanan birçok kişiye ulaşabilir. Bu çalışmanın ana motivasyonu da budur.

Kullanıcıları kümelendirmek YSA bazlı öğrenme kısmını geliştirmiştir. Bütün kullanıcıları aynı YSA’ya atamaktansa, açık yollarla elde edilen tercih edilen tür bilgisi kullanılarak kümeleme yöntemi kullanılmıştır. GA, kümeleme

seçeneklerinden biridir, fakat iyi bilinen K-ortalamalar kümeleme algoritması ile kıyaslandığında daha iyi bir yaklaşım olarak kendini kabul ettirmiştir.

GA ile kümelemede cezalandırıcı bir dönüşüm uygulamak, işlem maliyeti ve zamanı açısından önemli bir değişken olan YSA ağ boyutunu kontrol altında tutmaktadır. Benzer şekilde, YSA eğitimine rasgele değerler yerine önceden işlenmiş ağırlıklarla başlamak verimi arttırmaktadır.

Anahtar kelimeler: Yapay sinir ağları, küme sayısının otomatik olarak belirlenmesi, GAKK, kümeleme, veri toplama, sayısal alıcılar, genetik algoritmalar, KygTV, K-ortalamalar algoritması, çok katmanlı algılayıcı, program önerisi, öneri motoru, kullanıcı profilleme

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER ONE
INTRODUCTION

In old days, where there was limited number of channels on television, it was easy for the users to find what to watch. With the developing broadcasting technologies and enhancements in consumer electronics devices, the number of channels increased significantly. The analogue to digital transition enabled utilization of the bandwidth in a more efficient way, which resulted in users having access to more than thousands of channels. The increasing number of channels created the modern age problem for users: how to find the program that matches each user's taste within that many channels.

Recommending television programs based on user's viewing habits has already been studied for some time. Nevertheless, two main challenges, data collection and modelling the user, have not been fully addressed. Most commonly, data collection is limited to single device and single user and modelling is limited by the capacity of the receiver. Although receiver processors and memory units are improving year by year, they are not quite like computers. Considering the fact that servers have already evolved into highly capable devices that can handle big data operations as well as multiple requests from multiple clients, instead of performing all the calculations on the receiver side, it would be wiser to centralize the intelligence and the logic on the server side. Furthermore, broadcasters can access cloud services for extensive computing using a wide range of parallel computing possibilities when necessary.

Another drawback of current systems is the necessity of receiver software modifications. This means that the user should have a particular receiver or one within a small set of compatible receivers to have the benefits of these systems. Apparently, this limits the expansion of these recommendation systems among consumers. Users face the dilemma of getting the recommendation system or using their favorite brand.

For data collection part, in broadcasting environments, it is not normally possible to collect content consumption data from all users. Therefore, data are collected from a selected number of users in a “panel”. Alvarez et al. (2009) give United Kingdom as an example where panel members have a device connected to their receivers. This device collects information by identifying the program and channel automatically. They also note that usage of proprietary methods are not prepared for the advancing broadcasting era and can only be applied to certain use cases.

These challenges are partially addressed in the Internet Protocol Television (IPTV) environment, but they are limited to television signals delivered over IP networks. Users usually receive IPTV via a set-top box (STB), which is connected to broadband Internet, using a closed network infrastructure. This means that every IPTV system may have their proprietary STB software and a recommendation method designed for one cannot be used for the others.

HbbTV concept provides a perfect solution to the problems stated above since it gives the framework to store user’s choices on a dedicated server using standardized methods with no additional equipment at user side other than the receiver itself. Once the model is constructed at server side using the collected data from many users, viewers can get recommendations via HbbTV without the need for proprietary receiver software. With more than 30 million receivers already supporting HbbTV throughout the world (Illgner-Fehns, 2015), broadcasters can deliver recommendations to a wider audience.

In this thesis, a novel approach is proposed to recommend television programs to the users based on their viewing habits. In order to achieve this, Genetic algorithms (GAs), artificial neural networks (ANNs) and Hybrid Broadcast Broadband Television (HbbTV) technologies are brought together.

In the proposed approach, it is shown that the history of the user can be gathered from the consumer side by an HbbTV application and stored on the server side. However, broadcasting the HbbTV application to multiple users is beyond the focus of this study. Therefore, as there are no databanks publicly available having such TV viewing data, they are collected offline from multiple users to construct the model.

Users are asked to specify their age, gender, and program genre preferences for three different time of the week during the registration screen of the HbbTV application. After the data collection phase, the optimum number of clusters is found by a new method, Controlled Clustering with Genetic Algorithms (CCGA). Collected data are clustered into the smaller groups by the same method. Then, a separate ANN for each cluster is formed and trained by parallel processing. The initial weights of these ANNs are determined by GA. The time of the day, the day of the week, the age and gender of the user and program genres are the inputs and thirteen neurons (each representing a genre) are the outputs. Once the model is ready to make recommendations, the programs fitting the profile are presented to the users again by the HbbTV application.

The proposed method in this thesis differs from the existing methods by the utilization of the HbbTV concept instead of proprietary solutions. The idea of collecting user’s program watching history by an HbbTV application running on a digital receiver, using these data to cluster the users by CCGA and to train ANNs on the broadcaster’s server side, and sending programs matching to the ANN’s output as recommendations via the same HbbTV application is not done before to the best of author’s knowledge and gives successful results. This makes the proposed method applicable to all consumer electronics digital receivers regardless of their brands or manufacturers and the delivery system used; cable, satellite, terrestrial, IP, etc.

This thesis is organized as follows: Chapter Two provides theoretical background on the concepts used. Chapter Three gives an overview of the related work. The proposal is detailed in Chapter Four, having sections on data collection, CCGA, learning, and program recommendation. Performed experiments and obtained results are given in Chapter Five. Finally conclusions are drawn in Chapter Six.

CHAPTER TWO
THEORETICAL BACKGROUND

The followings are used in this thesis:

- HbbTV
- ANNs
- Genetic Algorithms for clustering
- K-means algorithm for comparison with GA clustering
- Genetic Algorithms for ANN weight initialization

Therefore, in this chapter theoretical information is given for each of them.

2.1 Hybrid Broadcast Broadband TV (HbbTV)

HbbTV stands for Hybrid Broadcast Broadband TV. It aims to combine traditional broadcasting with broadband content delivery in the same device (TV or STB), as described in Figure 2.1.

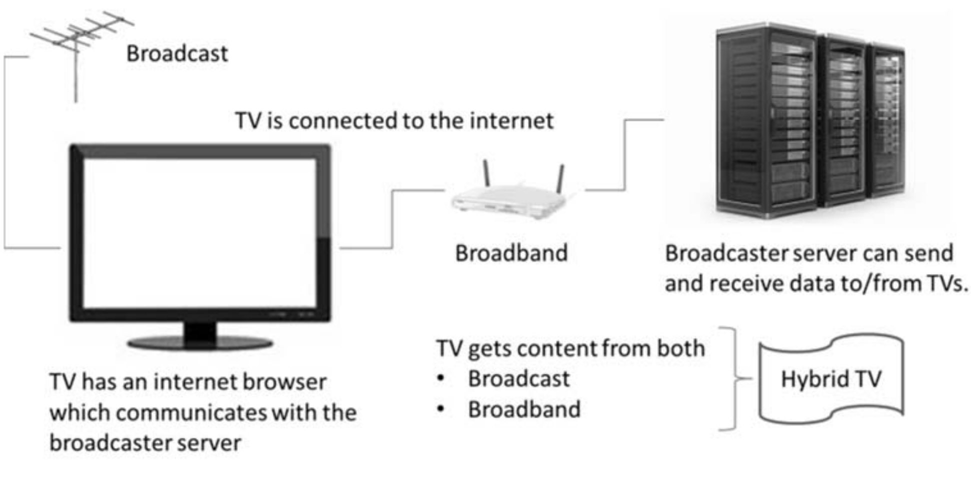

Figure 2.1 Hybrid Broadcast Broadband TV.

The HbbTV specification is prepared by major companies from the industry as a joint work. The technical specification is published by European

Telecommunications Standards Institute (ETSI) under the name TS 102 796 for harmonizing the broadcast and broadband delivery of entertainment services to consumers through digital receivers (ETSI, 2015). It aims to improve television watching experience for consumers by enabling innovative, interactive services over broadcast and broadband networks.

According to HbbTV specification, “HbbTV defines a platform for signaling, transport, and presentation of enhanced and interactive applications designed for running on hybrid terminals that include both a DVB compliant broadcast connection and a broadband connection to the Internet” (ETSI, 2016b, p. 14). HbbTV does not invent the wheel again; rather it uses the available technologies and combines them wisely (Figure 2.2).

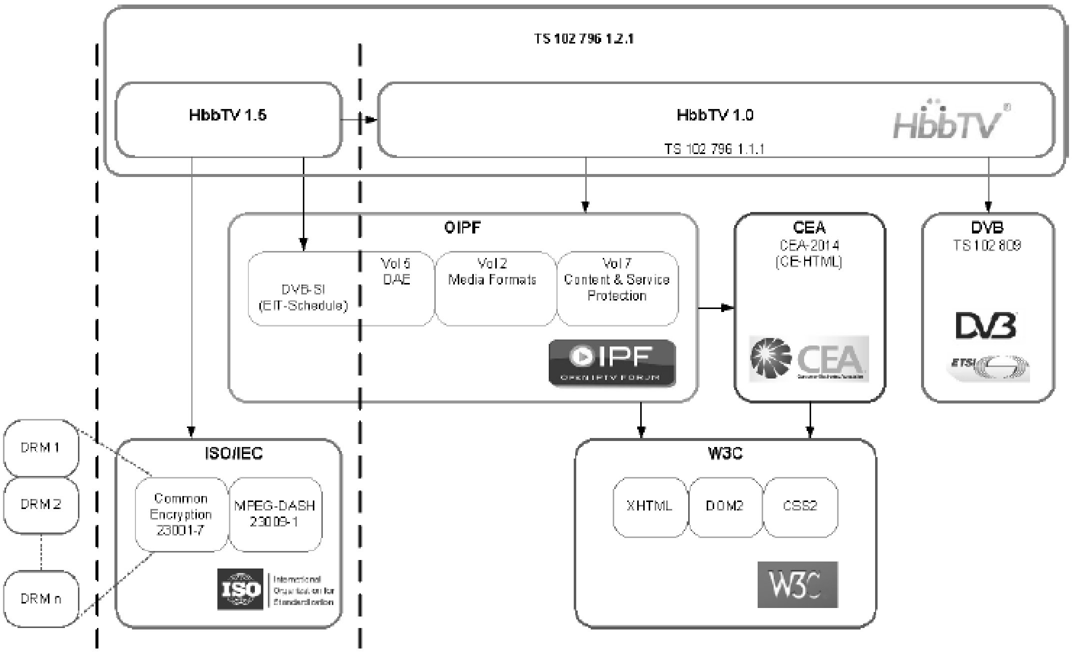

Figure 2.2 HbbTV specification overview (ETSI, 2015).

The most important technologies that HbbTV is based on are

- CE-HTML (CEA-2014 revision A: “Web-based Protocol and Framework for Remote User Interface on UPnP™ Networks and the Internet (Web4CE)”),

- OIPF (Open IPTV Forum Release 1 specification),
- Application Information Table (AIT) signaling (TS 102 809, “Signalling and carriage of interactive applications and services in Hybrid Broadcast Broadband environments”),
- DASH (MPEG DASH, “Information technology – Dynamic adaptive streaming over HTTP (DASH) -- Part 1: Media presentation description and segment formats”),
- CENC [MPEG CENC, “Information technology -- MPEG systems technologies -- Part 7: Common encryption in ISO base media file format files”).

An HbbTV application is a collection of HyperText Markup Language (HTML), JavaScript, Cascading Style Sheets (CSS), Extensible Markup Language (XML) and multimedia files constituting an interactive service. As of October 2014, HbbTV has a big momentum and interest in all over the world (Figure 2.3).

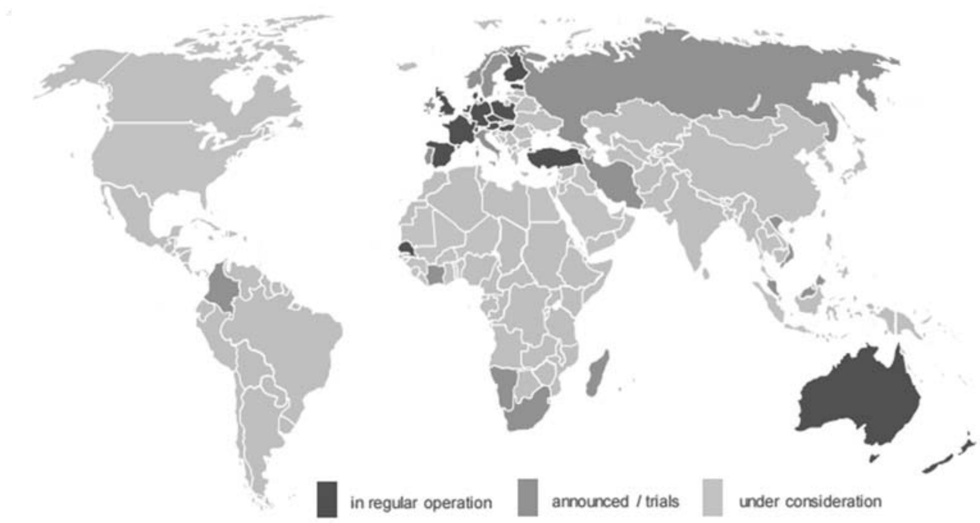

Figure 2.3 HbbTV world map (Illgner-Fehns, 2015).

The history and evolution of the HbbTV specifications are as follows:

- **ETSI TS 102 796 v.1.1.1 (06/2010)**

This is the original HbbTV specification.

- **ETSI TS 102 796 v.1.2.1 (11/2012)**
  TS 102 796 v.1.2.1 (often called HbbTV 1.5) introduced support for HTTP adaptive streaming (based on MPEG-DASH), which automatically selects the audio/video quality based on Internet connection speed. It also enabled content protection of DASH content with DRM technologies based on the MPEG CENC specification.
- **ETSI TS 102 796 v.1.3.1 (10/2015)**
  In this version (often called HbbTV 2.0), the Web platform is updated to HTML5 and a number of new features is introduced. It has been replaced by HbbTV 2.0.1.
- **ETSI TS 102 796 v.1.4.1 (08/2016)**
  The HbbTV 2.0.1 specification updates HbbTV 2.0 by some features needed for deployment in the UK and Italian market. It also fixes issues found during implementation of HbbTV 2.0 receivers and tests.

2.1.1 System Overview

As shown in Figure 2.4, an HbbTV capable device can connect to both digital video broadcasting (DVB) via broadcast network and Internet via broadband network. From the broadcast part (terrestrial, cable, or satellite), it receives traditional audio/video content (linear A/V content). HbbTV application signaling also comes from broadcast. From the broadband part, the device receives application data and on demand audio/video streaming (non-linear A/V content).

2.1.2 Functional Terminal Components

An HbbTV device has both broadcast interface and broadband interface (Figure 2.5). By demultiplexing the data coming from broadcast, the device gets the Uniform Resource Locator (URL) information of the HbbTV application. Then, the device goes to that URL via its Internet connection and shows the content on its browser. Linear and non-linear A/V content is shown by the media player component.

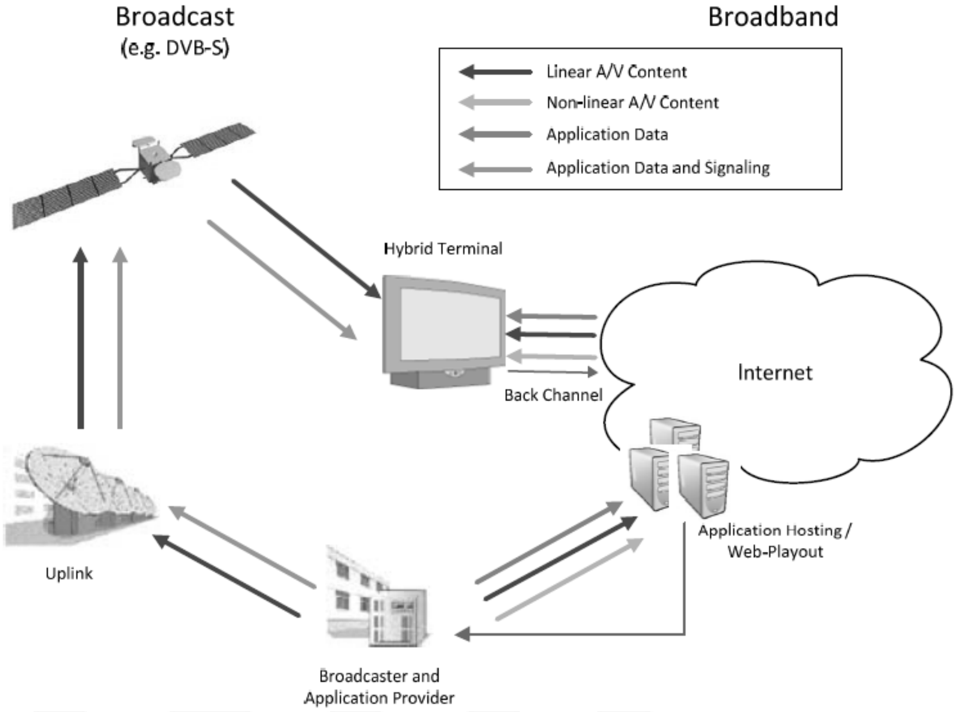

Figure 2.4 System overview (ETSI, 2015).

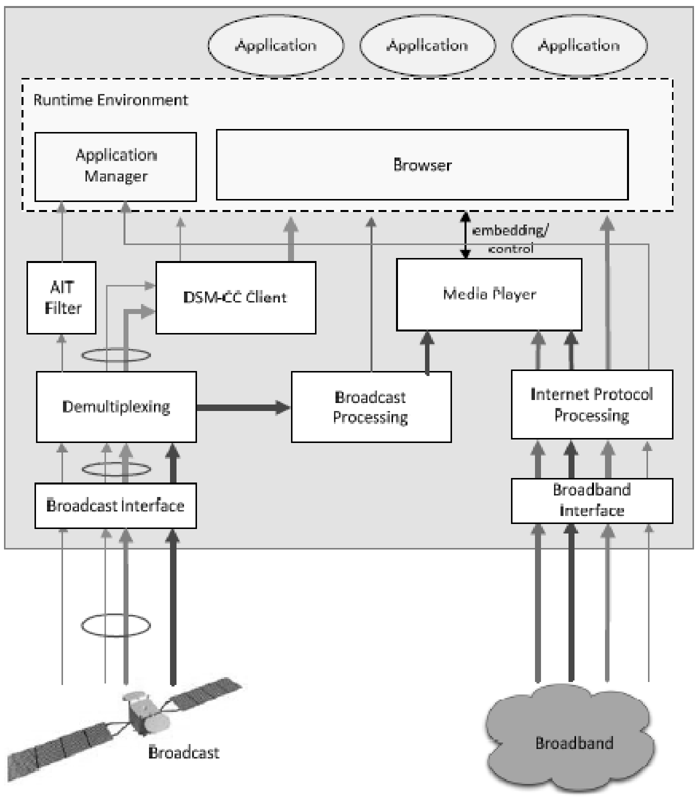

Figure 2.5 Functional terminal components (ETSI, 2015).

2.1.3 HbbTV Application Examples

An HbbTV application can be auto-launch, i.e., it can start automatically without any user interaction. They are usually designed as the red-button applications, to inform the user about the existence of the application (Figure 2.6).

Figure 2.6 Red-button example (ETSI, 2015).

A typical HbbTV application life cycle consists of below steps:

- User tunes to a channel, starts watching it,
- TV gets a URL from digital broadcast signal,
- TV browser connects to that URL,
- Red button appears on the bottom right corner, as an indication to the user that this channel has HbbTV application,
- User presses red button, TV browser goes to broadcaster’s HbbTV portal page,
- User navigates in the available applications,
- User exits the application by changing channel, etc.

The most popular HbbTV applications seen so far are catch-up content, video on demand, supplementary information on the current event, games, quizzes, voting, EPG (electronic program guide), weather forecast, etc. (Figure 2.7, Figure 2.8 (“Das Erste auf Ihrem Smart TV,” n.d.) and Figure 2.9 (“Overview,” n.d.)):

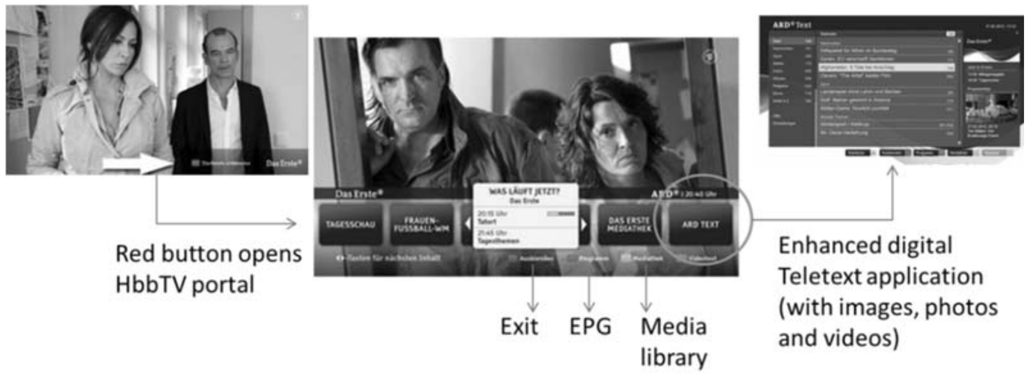

Figure 2.7 Typical HbbTV applications - 1.

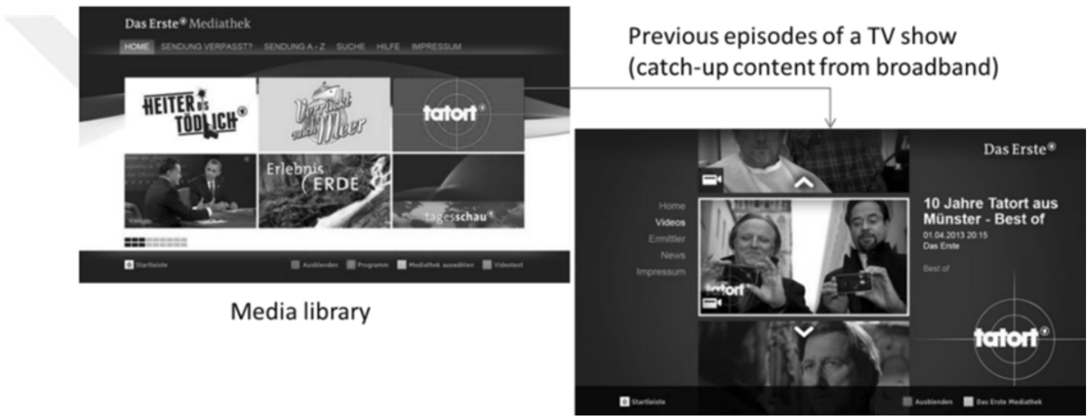

Figure 2.8 Typical HbbTV applications - 2.

Figure 2.9 Typical HbbTV applications - 3.

2.2 Artificial Neural Networks (ANNs)

The work on ANNs has been very promising in treating nonlinear process modelling. ANNs offer the possibility of studying large complex nonlinear systems. Such nonlinear systems may be based on data which is noisy or dependent on each other.

Main operating unit of ANNs is basic artificial neuron, which can be modelled as a nonlinear device with multiple inputs connected to other neurons by synaptic weights. These synaptic weights are updated according to an adaptive algorithm, which gives ANN the learning capability. By adjusting the synaptic weights, ANNs can produce desired outputs when appropriate inputs are applied (Kumluca, 1997).

Multilayer perceptron (MLP) is a large class of feedforward neural networks with neurons arranged in layers. In this structure, all neurons in adjacent layers are connected through uni-directional links called synaptic weights. Figure 2.10 shows a four-layer MLP structure at which all the nodes are named to be used in equations.

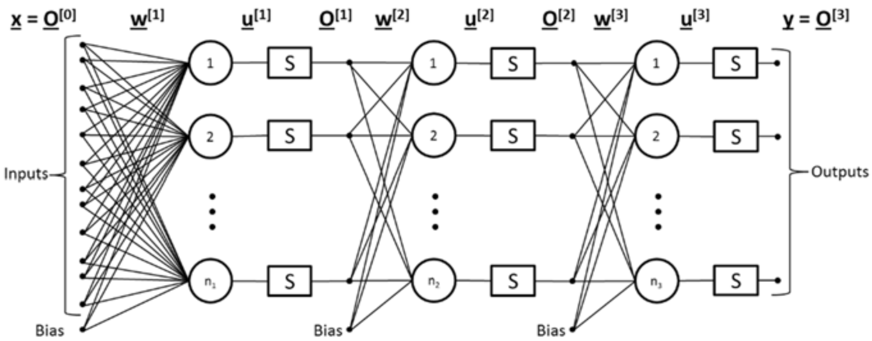

Figure 2.10 Architecture of the four-layer perceptron used in this work.

The weighted sum of the inputs at each layer is

$$\underline{u}^{[s]} = \underline{O}^{[s-1]} \underline{w}^{[s]} \quad s = 1, 2, 3 \tag{2.1}$$

where $\underline{w}^{[s]}$ is the synaptic weight matrix of the s^{th} layer and $\underline{O}^{[s\text{-}1]}$ is the input vector to that layer including the external threshold, also called an offset or bias.

The neuron output is computed by passing the weighted sum of its inputs by a nonlinear bounded activation function $\underline{\psi}^{[s]}$, i.e.,

$$\underline{O}^{[s]} = \underline{\psi}^{[s]}\left(\underline{u}^{[s]}\right) = \underline{\psi}^{[s]}\left(\underline{O}^{[s-1]}\underline{w}^{[s]}\right) \tag{2.2}$$

where it is usually determined by a monotonically increasing sigmoid (S-shaped) function of a weighted sum of the input signals. A possible sigmoid function may be defined as in the following equation:

$$y = \frac{1}{1+e^{-\gamma u}} \tag{2.3}$$

for an unsymmetrical unipolar representation where γ is a positive constant or variable, which controls the steepness of the sigmoidal function (Figure 2.11).

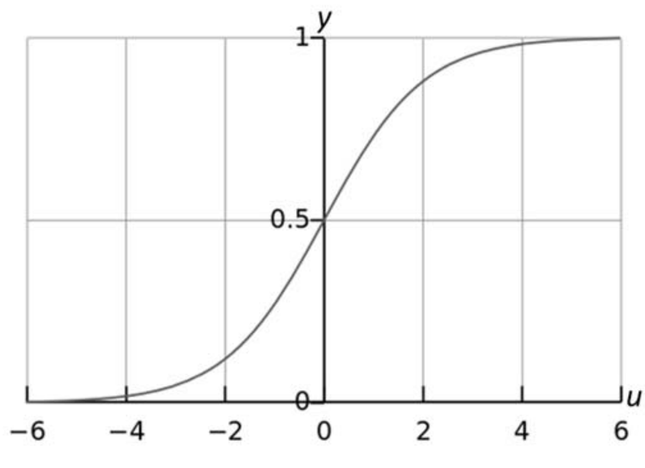

Figure 2.11 Sigmoid limiter for γ = 0.9.

Learning of the MLP consists of the adaptation of all synaptic weights in such a way that the discrepancy between the actual output signals $\underline{y}$ and the desired signals $\underline{d}$, averaged over all learning examples, is as small as possible. The back propagation learning algorithm can be considered as an unconstrained optimization problem of a suitably constructed error function (Cichocki & Unbehauen, 1993).

In the on-line learning technique, the training patterns are presented sequentially, usually in random order. The standard back propagation algorithm uses the "Steepest-Descent Gradient Approach" to minimize the mean-squared error function.

For each learning sample the weights $\underline{w}^{[s]}$ are changed by an amount $\underline{\Delta w}^{[s]}$ proportional to the respective negative gradient of the local error function *E*, which can be written as

$$E = \frac{1}{2}\sum_{j=1}^{n3}\left(d_j - y_j\right)^2 \tag{2.4}$$

$$\underline{\Delta w}^{[s]} = -\eta \frac{\partial E}{\partial \underline{w}^{[s]}} \tag{2.5}$$

where η is a sufficiently small positive learning parameter. Of a four-layer MLP, change in weights becomes

$$\underline{\Delta w}^{[s]} = \eta \underline{O}^{[s-1]T} \delta^{[s]} \tag{2.6}$$

where for the output layer with n_3 neurons

$$\delta_j^{[3]} = \left(d_j - y_j\right)\frac{\partial \psi_j^{[3]}}{\partial u_j^{[3]}} \quad j = 1,2,\ldots,n_3 \tag{2.7}$$

for the second hidden layer with n_2 neurons,

$$\delta_j^{[2]} = \frac{\partial \psi_j^{[2]}}{\partial u_j^{[2]}} \sum_{l=1}^{n3} \delta_l^{[3]} w_{jl}^{[3]} \quad j = 1,2,\ldots,(n_2 + 1) \tag{2.8}$$

and for the first hidden layer with n_1 neurons

$$\delta_j^{[1]} = \frac{\partial \psi_j^{[1]}}{\partial u_j^{[1]}} \sum_{l=1}^{n_2+1} \delta_l^{[2]} w_{jl}^{[2]} \qquad j = 1,2,\ldots,(n_1+1) \tag{2.9}$$

are found. It should be noted that

$$\frac{\partial \psi_j^{[s]}}{\partial u_j^{[s]}} = \gamma \psi_j^{[s]} \left(1 - \psi_j^{[s]}\right) \tag{2.10}$$

The weights for the next pattern are updated using the formula

$$\underline{w}^{[s]}(p+1) = \underline{w}^{[s]}(p) + \alpha \underline{\Delta w}^{[s]}(p) \tag{2.11}$$

where p is the current pattern and α is called the momentum term, which is added to smooth the weight changes in order to improve the standard back propagation learning algorithm.

2.3 Evolutionary Computation

Evolution is an adaptive process. The population adapts to the environment better by evolution. The individuals who can adapt to the environment will have better chance of survival by natural selection (survival of the fittest). Evolutionary algorithm is an iterative and stochastic process that operates on a population, which is composed of randomly generated individuals, each individual being a potential solution to the problem of concern. For each individual, a fitness value, which is showing the fitness of the solution to the problem, is calculated and this value is used as the quantitative information by the algorithm to guide the search (Sivanandam & Deepa, 2008).

Evolutionary computation (EC) techniques use these principles of evolution into algorithms that would lead to the optimal solutions to the problem. Unlike traditional search algorithms, which randomly sample (e.g., random walk) or heuristically sample (e.g., gradient descent) the search space one solution at a time (in search for the optimal solution), an evolutionary search algorithm uses an efficient directed search based on a population composed of a large number of individuals who are adapted in successive generations. This generally makes evolutionary search better than the traditional search algorithms and it is not susceptible to the hill-climbing behaviors of gradient-based search (Sivanandam & Deepa, 2008).

The importance of EC has increased in the last 20 years when compared with traditional search and optimization techniques. The flexibility and robustness of the EC techniques helped EC to be considered as an adaptable approach for complex optimization problems (Alba, 2013).

The advantages of EC include

- Conceptual simplicity,
- Broad applicability,
- Hybridization with other methods,
- Parallelism,
- Robust response to dynamically changing circumstances,
- Solves problems that have no solutions,
- Flexibility, and so on.

EC has different families as shown in Figure 2.12. They have data structures which encode a solution as one of the following forms:

- strings over a finite alphabet → Genetic Algorithms
- real valued vectors → Evolutionary Strategies
- finite state machines → Evolutionary Programming
- trees → Genetic Programming

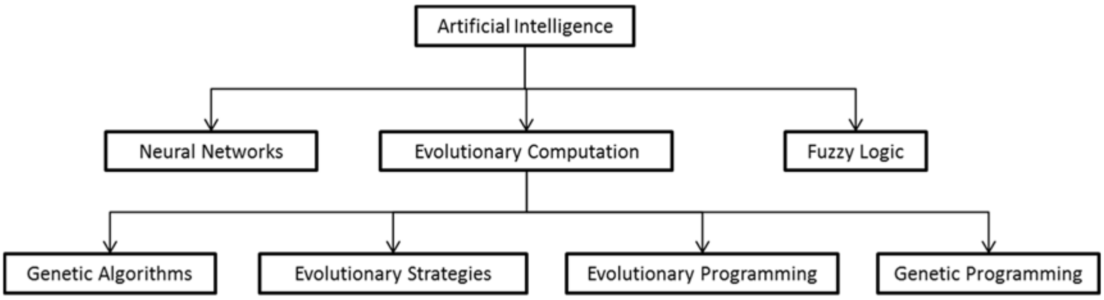

Figure 2.12 EC families (Sivanandam & Deepa, 2008).

The GAs can be considered as the most capable method among the evolutionary techniques representing the application of evolutionary tools. GAs are also considered to be the most widely used computation models of evolution in artificial intelligence systems (Sivanandam & Deepa, 2008).

2.4 Genetic Algorithms

GA's searching method is quite similar to the natural selection in nature. An "evolved" solution is tried to be found by applying the "survival of the fittest" principle. Possible solutions form a population and the fitter members of the population (which are nearer to the solution) are more likely to survive and pass their genes to the next generation. As the generations evolve, the population should come closer to the solution (Jea, 2000).

In GAs, the search space parameters are described in the form of strings (chromosomes). These strings form a population. At the beginning, the population is randomly formed representing different characteristics of the search space. For each string, the goodness of the string is calculated using a fitness function. Utilizing the survival of the fittest principle, a few of these strings are selected and mated. New generation of strings is formed using crossover and mutation, which are inspired from evolution. This process is run until the termination criteria is met (Maulik & Bandyopadhyay, 2000).

2.4.1 General Information

Steps of the GA can be given as below (Sivanandam & Deepa, 2008):

1. Define the problem and the fitness function,
2. Form an initial population consisting of individual solutions,
3. Evaluate each individual in the population against the fitness function,
4. Depending on their fitness function, select a number of individuals as parents,
5. Reproduce offspring from those parents by crossover and mutation,
6. Calculate offspring fitness functions,
7. Select survivors among parents and offspring and form the new population,
8. If the termination criterion is satisfied then stop; else go to step 3.

Above steps are visualized in the following flowchart (Figure 2.13).

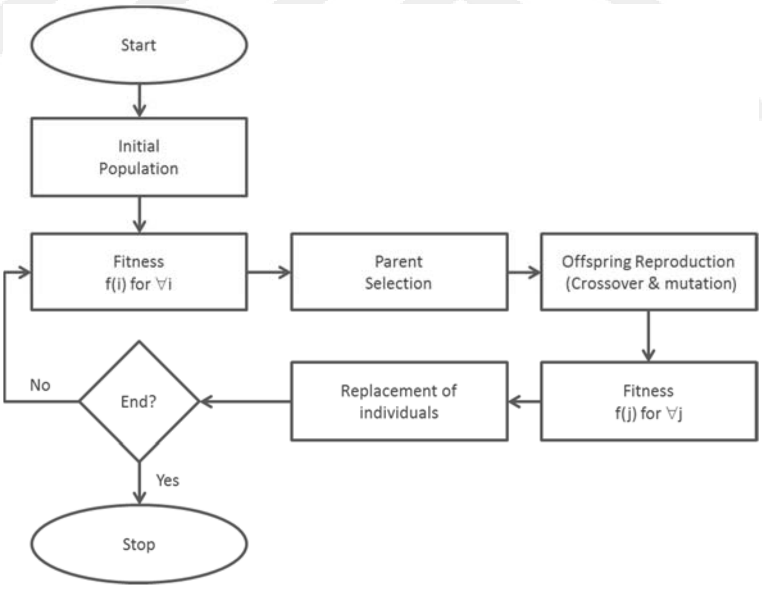

Figure 2.13 GA flowchart.

2.4.1.1 Terminology

As expected, GA takes its parameter names from the natural evolution terms. A nomenclature can be given as in Table 2.1.

Table 2.1 GA nomenclature

Natural Evolution	GA
Chromosome	String
Gene	Feature
Allele	Value of the feature
Locus	Position in string
Genotype	Coded string
Phenotype	Decoded value

A chromosome is a sequence of genes; and a gene is a bit of string of arbitrary length (Figure 2.14). Phenotypes are the objects forming possible solutions within the original problem context. Genotypes are the individuals within the GA (Sivanandam & Deepa, 2008). For example, integer 18 is a phenotype; binary coding of 18, 10010 is a genotype.

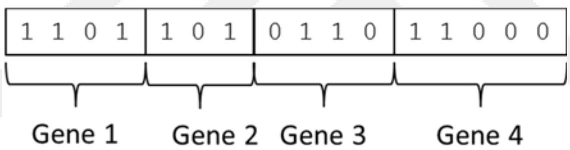

Figure 2.14 Chromosome consisting of genes.

2.4.1.2 Encoding

Encoding is the first design step. At this step, mapping from the phenotypes onto a set of genotypes that represent these phenotypes is performed (Figure 2.15).

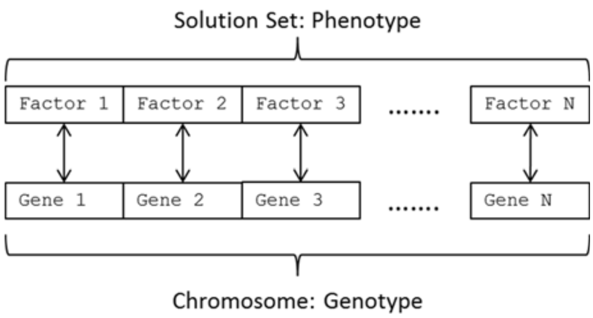

Figure 2.15 Mapping between phenotypes and genotypes.

Binary, Octal, Hexadecimal, etc. encoding methods are the examples of different encoding techniques (Figure 2.16).

Encoding	Binary	Octal	Hexadecimal
Chromosome 1	1 0 0 1 0 0 1 0 1 1 1 0	4 4 5 6	0 9 2 E
Chromosome 2	1 1 0 1 1 0 0 0 0 0 1 1	6 6 0 3	0 D 8 3

Figure 2.16 Binary, octal, and hexadecimal encoding examples.

2.4.1.3 Population

A population is a collection of individuals (Figure 2.17). Individuals are static objects that do not change or adapt, it is the population that does. For each and every problem, the population size will depend on the complexity of the problem. Often a random initialization of population is carried out. In almost all GA applications the population size is constant, not changing during the evolutionary search.

Population	Chromosome 1	3 E F A 0 2 9 C
	Chromosome 2	D 1 8 B 1 4 D A
	. . .	. . .
	Chromosome N	4 3 B A F 0 2 C

Figure 2.17 Population of chromosomes.

The size of the initial population should be as large as possible to cover the whole search space. However, as the population gets larger, the computational cost, memory, and time also increase (Sivanandam & Deepa, 2008).

2.4.1.4 Fitness

The fitness in GA is calculated by the fitness function for the individual. The fitness value is crucial as it forms the basis for mating pool selection. In order to calculate the fitness, the chromosome is first decoded and fitness function is evaluated. Higher fitness value shows the goodness of the solution as well as closeness of the individual to the optimal one (Sivanandam & Deepa, 2008).

2.4.1.5 Selection

The selection process selects chromosomes from the mating pool directed by the survival of the fittest concept of natural genetic systems. The purpose of selection is to emphasize fitter individuals in the population in hopes that their offsprings have higher fitness.

It distinguishes among individuals based on their quality and allows better individuals to become parents of the next generation. It is typically a probabilistic process. High quality individuals get a higher chance to become parents; low quality individuals are often given a small but positive chance (otherwise search could get stuck in a local optimum).

The following methods are seen to be used in this phase:

- Roulette-wheel selection
- Random selection
- Rank selection
- Tournament selection
- Boltzmann selection
- Stochastic universal sampling

Roulette-wheel selection is the common technique that implements the proportional selection strategy.

2.4.1.6 Reproduction

Reproduction means creating new individuals from old ones, i.e., generating new candidate solutions. After the reproduction process, the population is expected to be enriched with better individuals. Variation operators such as crossover and mutation constitute reproduction and outcome is called offspring.

2.4.1.6.1 Crossover. Crossover is a process where two offsprings are produced by two parent chromosomes exchanging genetic information with random probability (Maulik & Bandyopadhyay, 2000). A better offspring is expected to be created by the application of crossover operator to the mating pool (Sivanandam & Deepa, 2008).

There are quite a few ways to perform crossover. Some of them are:

- Single point crossover
- Two point crossover
- Multi-point crossover (N-point crossover)
- Uniform crossover
- Three parent crossover
- Crossover with reduced surrogate
- Shuffle crossover
- Precedence preservative crossover
- Ordered crossover
- Partially matched crossover
- Probabilistic crossover

For example, in single point crossover (Figure 2.18), for chromosomes of length *l*, a random integer, called the crossover point, is generated in the range [1, *l* – 1]. The portions of the chromosomes lying to the right of the crossover point are exchanged to produce two offsprings.

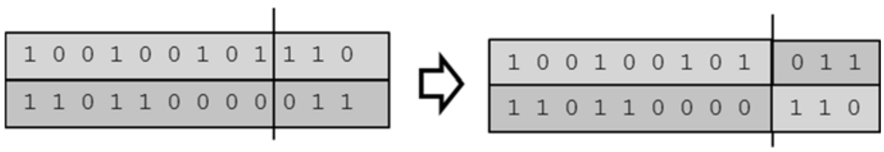

Figure 2.18 An example of single point crossover.

2.4.1.6.2 Mutation. Mutation is an important part of the GA as it provides means to avoid irreversible loss of information. It also creates new data by random modification of the genes which could prove vital to find the global minimum. Mutation prevents the algorithm to be trapped in a local minimum (Sivanandam & Deepa, 2008). In a sense, it can be regarded as filling the gene pool with fresh blood.

After crossover, each chromosome may undergo mutation with a small probability. For binary representation of chromosomes, a bit position (or gene) is mutated by simply flipping its value (Figure 2.19). For floating representation, value of the gene is altered by a uniformly distributed percentage.

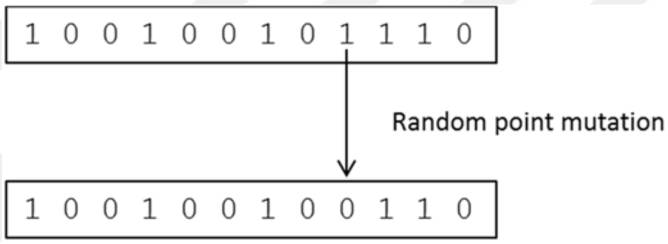

Figure 2.19 Mutation on binary representation.

2.4.1.7 Replacement

Replacement is the last stage of the breeding cycle. It is also called Survivor Selection Mechanism. Since the population size is kept constant, once offsprings are produced, a method must determine which of the current members of the population, if any, should be replaced by the new solutions. Replacement is often deterministic and based on individual fitness values or ages (in favor of offspring).

2.4.1.8 Termination

As GAs are stochastic and there is no guarantee to reach a global optimum value, termination criterion is determined by either one of the following methods as defined by Sivanandam and Deepa (2008):

- Best individual: terminates when the minimum fitness in the population is less than the convergence value.
- Worst individual: terminates when the least fit individuals of the population have fitness below the convergence criteria.
- Sum of fitness: terminates when total fitness of the entire population is less than or equal to the convergence value.
- Median fitness: terminates when most of the individuals have higher fitness than the convergence value.
- Computation time: terminates when the time allocated for the algorithm to run expires.
- Iteration number: terminates when the algorithm runs for a predefined number of cycles.

2.4.1.9 Comparison and Limitations

Almost all traditional search algorithms perform search from a single point while GAs operate on all population, i.e., GAs use all available solutions rather than a single solution. This makes GAs quite robust. It also increases the chance of finding the global optimum while avoiding to be stuck in local optima as shown in Figure 2.20 (Sivanandam & Deepa, 2008).

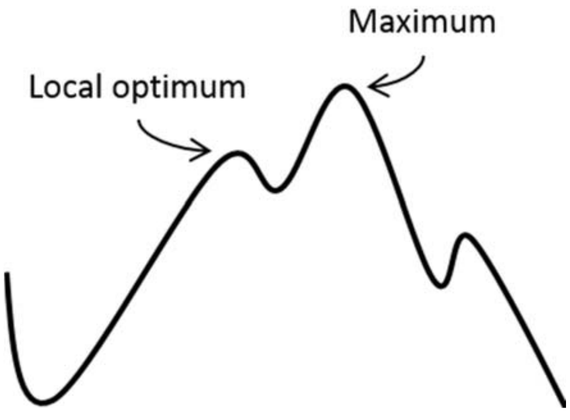

Figure 2.20 GA vs. hill climbing.

However, this does not guarantee GAs to find the global optimum solution to the problem of concern. Finding an acceptable solution would be satisfactory for GAs (Sivanandam & Deepa, 2008). Working with a whole population of solution is very useful and increases the chance to find a global optimum but it is very expensive in terms of memory and operating time (Alba, 2013).

2.4.2 GA for Clustering

GA is a widely used method for data clustering. It can be applied to divide the data to a pre-defined number of clusters, as well as to detect number and elements of clusters automatically. Following sections give details on both applications.

2.4.2.1 Using Pre-defined Number of Clusters

In this case it is assumed that the data are to be clustered into a pre-defined number of sets. The population *P* consists of *N* chromosomes which is shown as:

$$P = \{K_1 \quad K_2 \quad \ldots \quad K_N\} \tag{2.12}$$

The cluster set is:

$$\{C_1 \quad C_2 \quad \ldots \quad C_M\} \tag{2.13}$$

where *M* is the number of clusters which is a pre-defined value.

Initial cluster centers are chosen randomly.

$$CC_i = K_j \quad i \in [1, M],\ j \in [1, N] \tag{2.14}$$

Then, the cluster center set can be represented as:

$$\{CC_1 \quad CC_2 \quad \ldots \quad CC_M\} \tag{2.15}$$

After determining the centers, chromosomes can be distributed to the clusters based on their distances. For each chromosome, the distance to each cluster center is calculated and the chromosome is assigned to the cluster with the nearest center.

These distances between the chromosomes and the cluster centers can be formulated as:

$$D_{ij} = \sum_{x=1}^{L} \left(K_j(x) - CC_i(x)\right)^2 \quad i \in [1, M],\ j \in [1, N] \tag{2.16}$$

where L is the length of the chromosomes. Then, for all K_j, i_{min} is found where

$$D_{i_{\min} j} = \min_i D_{ij} \tag{2.17}$$

K_j becomes a member of $C_{i_{min}}$

$$K_j \in C_{i_{\min}} \tag{2.18}$$

For example, clusters can consist of following chromosomes:

$$\begin{aligned} C_1 &= \{K_3 \quad K_{26} \quad \ldots \quad K_{208}\} \\ C_2 &= \{K_{11} \quad K_{43} \quad \ldots \quad K_{199}\} \\ &\ldots \\ C_M &= \{K_7 \quad K_{33} \quad \ldots \quad K_{88}\} \end{aligned} \tag{2.19}$$

With the chromosomes assigned to the clusters, centers are no more the real centers of the clusters. They need to be moved to the center of the new members:

$$CC_i(x) = \frac{1}{n_i} \sum_{j=1}^{n_i} K_j(x), i \in [1, M], x \in [1, L] \tag{2.20}$$

where n_i is the number of chromosomes in the i^{th} cluster.

The new cluster center set becomes:

$$\{CC_1 \quad CC_2 \quad \dots \quad CC_M\} \tag{2.21}$$

At this point, two cluster centers will be selected as parents to produce offsprings. However, parents need to be strong members of the population, according to the "survival of the fittest" phenomenon. Therefore, fitness values of each cluster are to be found.

The fitness function is the inverse of the sum of the distances of each cluster members to the center of that cluster; and this fitness function should be maximized.

The sum of distances for each cluster is:

$$D_i = \sum_{j=1}^{n_i} \sum_{x=1}^{L} \left(K_j(x) - CC_i(x)\right)^2 \quad i \in [1, M] \tag{2.22}$$

Therefore, the fitness value is:

$$F_i = \frac{1}{D_i} \quad i \in [1, M] \tag{2.23}$$

Then, the fitness values set becomes

$$\{F_1 \quad F_2 \quad \dots \quad F_M\} \tag{2.24}$$

Potential parents are the centers of the clusters. Each of them has a chance of selection depending on the fitness value of their clusters. The probability Φ_i of each cluster center against the fitness value F_i can be calculated as Roulette-wheel selection:

$$\Phi_i = \frac{F_i}{\sum_{j=1}^{M} F_j} \quad i \in [1, M] \tag{2.25}$$

where the larger fitness value, the better chance for selection.

Then, the set of cluster center probabilities becomes

$$\{\Phi_1 \quad \Phi_2 \quad \ldots \quad \Phi_M\} \tag{2.26}$$

The parents are chosen within the cluster centers based on their probabilities calculated in the previous step. The cluster center with higher probability is more likely to be chosen as parent. The first parent is CC_{i_1} with

$$\Phi_{i_1} = CC_{i_1}, i_1 \in [1, M] \tag{2.27}$$

and the second parent is CC_{i_2} with

$$\Phi_{i_2} = CC_{i_2}, i_2 \in [1, M] \text{ where } i_2 \neq i_1 . \tag{2.28}$$

Offspring production starts with crossing over the parent chromosomes at a random point. After crossover, offsprings go through mutation process with a low probability in order to prevent algorithm to be stuck in a local minimum. Then, the two parents (cluster centers) chosen previously are replaced with the produced offsprings. If the cluster centers are modified, then the algorithm continues from forming the clusters. If there is no change, it can be terminated.

A 2D example can be given to visualize how this GA clustering algorithm works. It is shown in Figure 2.21.

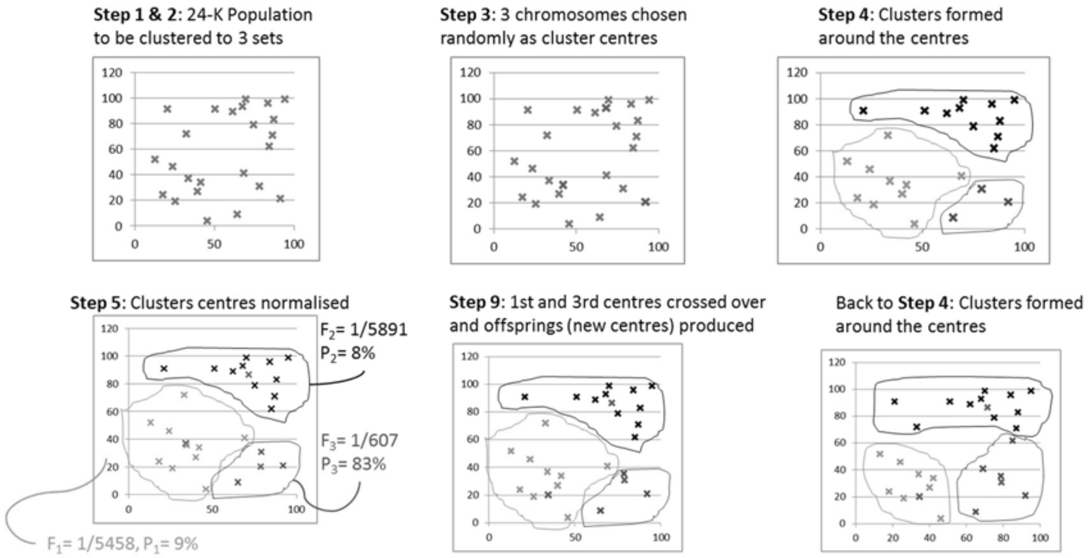

Figure 2.21 2D GA clustering example.

2.4.2.2 Automatic Detection of Number and Elements of Clusters

The method given by Bandyopadhyay and Maulik (2002) proposes automatic evolution of clusters and their numbers using GA. In this approach, there is a data set consisting of the vectors

$$DataSet = \{X_1 \quad X_2 \quad \ldots \quad X_D\} \tag{2.29}$$

and a population of chromosomes

$$P = \{K_1 \quad K_2 \quad \ldots \quad K_N\} \tag{2.30}$$

The minimum and the maximum number of centers are defined as K_{min} and K_{max}. Then the population size N is calculated as

$$N = 3(K_{max} - 1) \tag{2.31}$$

For each chromosome K_j, j = 1, …, N, a random number r_j (3, for example) is assigned. This means the chromosome K_j consists of r_j data points and (K_{max} – r_j) don't cares. For example, the following can be given

$$K_j = \{X_7 \quad \# \quad \# \quad X_{66} \quad \# \quad ... \quad \# \quad X_{181}\} \tag{2.32}$$

where X_7, X_{66}, and X_{181} are the initial cluster centers.

Then, the clusters are formed for each chromosome K_j by assigning data points to r_j clusters corresponding to the closest center. Cluster centroids are found as

$$z_i = \frac{1}{n_i} \sum_{X \in C_i} X \quad i = 1,..., r_j \tag{2.33}$$

where n_i is the number of data points in cluster C_i.

The scatter within C_i for each chromosome K_j is calculated as

$$S_i = \frac{1}{n_i} \sum_{X \in C_i} \|X - z_i\| \quad i = 1,..., r_j \tag{2.34}$$

and the distance between clusters C_i and C_k is

$$d_{ik} = \|z_i - z_k\| \quad i, k = 1,..., r_j \tag{2.35}$$

The Davies–Bouldin (DB) index for each chromosome K_j is measured to find the fitness value

$$DB_j = \frac{1}{r_j} \sum_{i=1}^{r_j} \max_{k,k \neq i} \frac{s_i - s_k}{d_{ik}} \tag{2.36}$$

$$F_j = \frac{1}{DB_j} \tag{2.37}$$

After calculating the fitness values, chromosomes are undergone crossover and mutation according to Roulette-wheel of DB_j. Offsprings are generated and replaced with old generation. This procedure is repeated with the new population and K_j with the best fitness is selected. Final number of clusters is equal to the number of data points in K_j. Final members of clusters are assigned around these data points (cluster centers).

2.4.3 GA for ANN Weight Initialization

In this case, the population P consists of N chromosomes

$$P = \{K_1 \quad K_2 \quad \ldots \quad K_N\} \tag{2.38}$$

where each chromosome represents a weight solution combined for the all MLP layers.

$$K_j = [w^{[1]} \quad w^{[2]} \quad w^{[3]}]_j \quad j = 1,2,\ldots,N \tag{2.39}$$

Chromosomes are randomly initialized within [-0.5, 0.5] interval. Then, the ANN output and local error E are calculated for each chromosome over the data set.

$$E = \{E_1 \quad E_2 \quad \ldots \quad E_N\} \tag{2.40}$$

Chromosomes producing smaller errors are more successful than the others. In terms of GA terminology, the fitness function becomes the local error. The GA steps are repeated as detailed in Section 2.4.1.

2.5 K-means Algorithm for Clustering

K-means is one of the simplest and widely used unsupervised learning algorithms. The difference between GA and K-means clustering is that there are no fitness calculation, crossover and mutation steps in the K-means. Objective is to minimize inter-cluster sum of squares function f

$$f = \sum_{i=1}^{M}\sum_{j=1}^{n_i}\sum_{x=1}^{L}\left(K_j(x) - CC_i(x)\right)^2 \tag{2.41}$$

where M is the number of clusters, n_i is the number of chromosomes in the i^{th} cluster, L is the length of the chromosomes, K_j is the j^{th} chromosome in the i^{th} cluster, CC_i is the cluster center of the i^{th} cluster. The algorithmic steps can be given as follows:

Step 1. Define M centroids, one for each cluster.

Step 2. Take each point belonging to a given data set and associate it to the nearest centroid.

$$D_{i_{\min} j} = \min_i D_{ij} \tag{2.42}$$

$$D_{ij} = \sum_{x=1}^{L}\left(K_j(x) - CC_i(x)\right)^2 \quad i \in [1, M],\ j \in [1, n_i] \tag{2.43}$$

Step 3. Re-calculate new centroids from the previous step.

$$CC_i(x) = \frac{1}{n_i}\sum_{j=1}^{n_i} K_j(x) \quad i \in [1, M], x \in [1, L] \tag{2.44}$$

Step 4. Stop if centers do not change. Otherwise, go to Step 2.

CHAPTER THREE
LITERATURE SURVEY

In this chapter, the previous works available in the literature, which are somehow related to this thesis, are summarized. Unlike the method proposed in this thesis, none of these works mentioned below has used a worldwide specification like HbbTV for data gathering or program recommendation. Moreover, some of them need proprietary software or additional hardware modules in the receiver or a PC tool for user registration and data gathering, which are not necessary for the method presented here.

3.1 User Profiling

There are many researchers working on the user profiling problem since the beginning of 2000s. Ardissono, Kobsa and Maybury (2004) summarize the problem as below:

> Given the heterogeneity of TV viewers, who differ e.g. in interests and skills, the provision of personalized services seems to be the only solution to address the information overload problem in an effective manner. The User Modelling and the Intelligent User Interfaces communities have, therefore, focused on the following main lines of research:
>
> - The provision of Electronic Program Guides recommending TV programs on an individual basis, to prevent users from "being lost in TV program space".
> - Information retrieval tools to help users select interesting content in the cases where a priori categorization of the content is not possible (e.g., in news shows).
> - The design and development of tools that help users explore large amounts of broadcast television content.

- The provision of adaptive interactive content that can be presented in a personalized way, depending on the viewer's interests.
- The design of suitable user interfaces that enable TV viewers to perform advanced tasks in an intuitive and efficient manner, which is essential for rendering Digital TV usable by any type of viewer, and not merely technical pundits (Ardissono, Kobsa et al., 2004, p. viii).

In recent years, television program recommendation systems and personalized program guides are the focus of many studies. Adomavicius and Tuzhilin (2005) present an overview of recommender systems, which can be classified into three main groups:

- content-based, where the users are recommended items similar to the ones they preferred in the past;
- collaborative, where the users are recommended items that other people with similar tastes and preferences liked in the past; and
- hybrid approaches where content-based and collaborative methods are combined.

In order to overcome the limitations of content-based and collaborative filtering systems, Barragáns-Martínez et al. (2010) propose a hybrid approach, which utilizes singular value decomposition to recommend television programs. Their solution is Web-based and users have to enter their preferences and ratings to their system via PC.

The TV Advisor of Das and Horst (1998), PTV system of Cotter and Smyth (2000), EPG work of Ardissono, Gena et al. (2004), work of Gutta et al. (2000), and method of Kurapati, Gutta, Schaffer, Martino and Zimmerman (2001) are examples of the earliest TV recommender systems. For a history of the evolution of recommendation engines, Smyth and Cotter (2004) can be referred.

The TV Advisor of Das and Horst (1998) expects explicit actions from TV viewers to generate recommendations. Such actions require individual users to explicitly specify their preferences in order to get high quality recommendations. Although this method is easy to implement in a digital receiver, it troubles users who want minimal interaction with the recommender. Furthermore, their system is not dynamic and does not allow evolution of user profiles over time.

Cotter and Smyth’s PTV uses a mixture of content-based reasoning and collaborative filtering in order to learn user preferences for generating recommendations (Cotter & Smyth, 2000). At the beginning, users state their interests about channel, genre, and viewing time while registering with the system, similar to the explicit recommender of Das and Horst (1998). However, the difference is that the PTV recommender learns user preferences as they enter their feedback on TV programs they have watched. Moreover, PTV is Internet based: it requires the users to log on to a Web site in order to see their recommendations. This approach may not be useful in the real setting as it removes users from the TV viewing environment.

Collaborative filtering (Balabanović & Shoham, 1997; Billsus & Pazzani, 1998), recommends TV programs that other users, having similar characteristics to a given profile, liked. The advantage of collaborative filtering is that it does not need content descriptions. However, it has the problem of not protecting users’ privacy since information about their likes and dislikes is used to make recommendations to other users. Moreover, collaborative filtering cannot work for programs that are completely new, not known to at least one of the viewers. This often happens with TV when new programs are broadcast.

Ardissono, Gena et al. (2004) create the Personalized EPG that employs a module-based system designed for digital receivers. Three user modelling modules collaborate in preparing the final recommendations: Explicit Preferences Expert, Stereotypical Expert, and Dynamic Expert. The Explicit Preferences Expert gets preferences declared by the users during initial setup. The Stereotypical Expert utilizes users' personal data known to the system and the explicit preferences stated in order to classify individual users into one of the stereotypical groups. The Dynamic Expert analyses users' watching behaviors and builds and adapts its model of the user.

There are works on both Internet connected and unconnected devices. Bjelica (2010, 2011), Krstic and Bjelica (2012) and Isobe, Fujiwara, Kaneta, Morita, and Uratani (2005) focus on receivers with no return channel, which requires all the logic and computation to be implemented in the receiver. Lee, Lee and Lee (2010) propose a personalized television program recommendation system under a cloud computing environment. But how the receiver communicates with the cloud is proprietary and their solution is limited to the receivers having this implementation. The work of Zhang, Zheng and Yuan (2005) also proposes a personalized television system running on standalone receiver compliant with Multimedia Home Platform (MHP) model, where results are given only for one user. However, they use some private application programming interfaces (APIs) in the receiver software.

The user profiling method proposed in this thesis is based on HbbTV. There is no other user profiling work utilizing HbbTV to the best of author's knowledge. Since it is a public standard and it does not require any special software or hardware, it can be realized wider than the above methods.

3.2 HbbTV

HbbTV is a relatively new standard. Therefore there are only a few scientific studies and publications about it. The most recent publications are the early results of this thesis (Topalli & Kilinc, 2016a, 2016b).

Other than that, Fondevila Gascon (2012) performed a survey among his journalism and communication students in Barcelona, Spain, and asked them about the perception of the HbbTV initiative. The results show that HbbTV (Internet & TV on-screen) is well received (value of 3.95 out of 5.00) among people having some technological knowledge. There are few papers focusing on the security and privacy issues associated with HbbTV. Oren and Keromytis (2014) argue that the broadband and broadcast systems are combined insecurely; they present that a realistic attack on smart TVs using low budget RF equipment is possible, and devices are vulnerable. Ghiglieri and Tews (2014) underline the fact that broadcasters and even neighbors can measure –without notice– consumers viewing behavior using HbbTV and this would violate their privacy rights. The latest HbbTV specification (ETSI, 2016b) has improvements which consider these criticisms.

3.3 IPTV

There are interesting works on IPTV systems in the literature. Hu et al. show how a network management system automatically collects IPTV-related data from STBs and provide audience rating to show how many customers are watching the specific channel within the specific time period (Hu et al., 2010). As common to IPTV operations, customers get their configured STBs from the IPTV provider, which directly manages STBs via the internal IPTV network. Therefore, their proposal is not suitable for the standard DVB receivers. Furthermore, they do not discuss how to use the collected user statistics.

The paper of Jabbar, Jeong, Hwang and Park (2008) presents an RFID (Radio Frequency Identification) application to identify and authenticate the viewer for personalized and interactive services offered by the IPTV. Their proposal include an RFID user tag programmed with certain rights for each individual, an RFID reader communicating with the tag wirelessly and connected with IPTV STB via RS-232 serial port. It is expected that the STB identifies the reader and loads its driver. Whenever the viewer is identified by the STB, the user information file is transferred from the server to the STB and personalized menus are shown. The communication of the STB with the IPTV server is proprietary.

The work done by Alvarez et al. (2009) describes an audience measurement model for the broadcasting and IPTV platforms. For DVB-H mobile phone case, they introduce a metering software module on the terminal which gathers, stores and transmits the service consumption data. For DVB case, they implement a set of software components running on top of a MHP stack to gather all the consumption measures. For IPTV platform, they distribute metering functionalities between the head-end and client side. Their proposal is based on proprietary software that needs to be installed on the receivers.

Yang, Park, Lee and Choi (2015) propose a Web-based IPTV content syndication system, which generates personalized program guide to provide a list of IPTV contents with respect to user interests and statistics of their online social community. In order to do this, users have to log in the Web site and send user metadata by explicitly notifying their preferences.

An automatic recommendation system of television program contents for IPTVs and conventional televisions using collaborative filtering is proposed by Kim, Pyo, Park and Kim (2011). However, they do not mention about data collection from conventional televisions. They only point out that on IPTV environments, user data can be collected on head-end side.

3.4 Genetic Algorithms

Over the years, GAs have been applied to many areas. Some of them can be listed as (Sivanandam & Deepa, 2008):

- Clustering
- Combinatorial optimization–set covering, traveling salesman problem, sequence scheduling, routing, bin packing, graph coloring and partitioning
- Control – gas pipeline, pole balancing, missile evasion, pursuit
- Design – semiconductor layout, aircraft design, keyboard configuration, communication networks
- Evolving LISP programs (genetic programming)
- Finding shape of protein molecules

- Functions for creating images
- Machine learning – designing neural networks, both architecture and weights, improving classification algorithms, classifier systems
- Nonlinear dynamical systems – predicting, data analysis
- Robot trajectory planning
- Scheduling – manufacturing, facility scheduling, resource allocation
- Signal processing – filter design
- Strategy planning

In this work, GA is used for clustering as detailed in Chapter Four. Sheikh, Raghuwanshi and Jaiswal (2008) present an overview of GA based clustering algorithms as below:

> Cluster analysis is a technique, which is used to discover patterns and associations within data. More specifically, it is a multivariate statistical procedure that starts with a data set containing information on some variables and attempts to reorganize these data cases into relatively homogeneous groups. One of the major problems encountered by researchers, with regard to cluster analysis that different clustering methods can and do generate different solutions for the same data set. What is needed is a technique that has discovered the most "natural" groups in a data set (Sheikh et al., 2008, p. 315).

Krovi (1992) was one of first to investigate the potential feasibility of using GAs for the purpose of clustering in his research. Krishna and Murty (1999) propose a novel hybrid GA to find a globally optimal partition of a given data into a specified number of clusters. They use Genetic K-means Algorithm (GKA), instead of crossover. Using finite Markov chain theory, they prove that the GKA converges to the global optimum. Moreover, GKA searches faster than some of the other evolutionary algorithms used for clustering.

In their work, Lu, Lu, Fotouhi, Deng, and Brown (2004a) present Fast Genetic K-means Algorithm (FGKA), which was inspired by GKA but features several improvements. Their experiments indicate that, while K-means algorithm might converge to a local optimum, both FGKA and GKA always converge to the global optimum eventually but FGKA runs much faster than GKA.

Incremental Genetic K-means Algorithm (IGKA), again by Lu, Lu, Fotouhi, Deng, and Brown (2004b), is an extension to previously proposed FGKA. IGKA inherits the striking feature of FGKA of always converging to the global optimum and it outperforms FGKA when the mutation probability is small.

Clustering Genetic Algorithm (CGA) is proposed by Kudová (2007). It outperforms the K-means algorithm on some tasks. In addition, it is capable of optimizing the number of clusters for tasks with well-formed and separated clusters. The framework is the same as in GA, while the individual building blocks of the algorithm are modified and adopted for the clustering task.

The GA-clustering proposed by Maulik and Bandyopadhyay (2000) uses searching capability of GAs for the purpose of appropriately determining a fixed number of cluster centers. The clustering metric that has been adopted is the sum of the Euclidean distances of the points from their respective cluster centers. The chromosomes, which are represented as strings of real numbers, encode the centers of clusters. Under limiting conditions, a GA-based technique also provides an optimal clustering with respect to the clustering metric being considered.

Lin, Yang and Kao (2005) propose GA-based unsupervised clustering that selects cluster centers directly from the data set, allowing it to speed up the fitness evaluation by constructing a look-up table in advance. They use binary representation rather than string representation to encode cluster centers.

Variable string length GA is used for developing a novel nonparametric clustering technique when the number of clusters is not fixed a priori (Bandyopadhyay & Maulik, 2001). Chromosomes are encoded in real number, and they have different lengths since they encode different number of clusters. The crossover operator is redefined to tackle the concept of variable string length. Cluster validity index is used as a measure of the fitness of a chromosome.

Bandyopadhyay and Maulik (2002) exploit the searching capability of GAs for automatically evolving the number of clusters as well as proper clustering of any data set. A new string representation, comprising both real numbers and the do not care symbol, are used in order to encode a variable number of clusters. The Davies–Bouldin index is used as a measure of the validity of the clusters. Effectiveness and utility of the genetic clustering scheme is demonstrated for a satellite image of a part of Calcutta. The proposed technique is able to distinguish some characteristic land cover types in the image.

In this work, a GA based algorithm CCGA is proposed to find the number and members of user clusters. It is a modified version of the method mentioned above (Bandyopadhyay & Maulik, 2002) and it introduces a penalizing transformation for the fitness calculation as explained in Section 4.2. This modified version is more appropriate for the time and size critical cases as the number of ANNs and the number of learning epochs need to be controlled in the problem presented here.

3.5 Genetic Algorithms and ANNs

A good deal of biological neural architecture is determined by genetic factors. Hence, it is not surprising that ANN researchers have been exploring how GAs can be organized in this area (Whitley, 1995). There are three main ways that GAs are used in conjunction with ANNs:

- to set the initial weights in fixed ANN architectures;
- to learn ANN topologies; and
- to select learning data and to interpret the output behavior of ANNs.

It is shown that using GAs to find a set of initial weights before applying gradient based methods may be advantageous for supervised learning classification problems

(Maulik & Bandyopadhyay, 2000). Therefore the weight matrices of the ANNs used in this thesis, are initialized by GA as detailed in Section 4.3.

3.5.1 GA for ANN Weight Initialization

GAs are successful in training feedforward networks. There are many works in the literature that use GA to set the learning and momentum rates for feedforward ANNs (Belew, Mcinerney, & Schraudolph, 1990; Harp, Samad & Guha, 1990; Schaffer, Caruana & Eshelman, 1990). Such settings are often done together with weight initialization. Montana and Davis (1989) run experiments on data from a sonar image classification problem. They illustrate the improvements gained by using GA in ANN weight initialization and witness the evolution of performance as they add more knowledge to the system.

3.5.2 GA for ANN Topology

Whitley, Starkweather and Bogart (1990) show that GA can be used to find network topologies that consistently display improved learning speeds over the typical feedforward ANNs. However, optimizing ANN architecture has a high cost in terms of evaluation time, which is so long, that it is not practical to use GA except for small topologies.

3.5.3 Genetic Algorithms for Pre-processing and Interpreting Data

In Chang and Lippmann’s work (1990), the GA reduces the input set from 153 to 33 input features without significantly changing the classification behavior. In Brill, Brown and Martin’s paper (1992), input set is also reduced; and, inputs that work well for a counter-propagation network are identified. Eberhart and Dobbins (1991) use a GA in their study to search for a decision surface that identifies boundary cases of appendicitis as predicted by an ANN. They search the input space for strings that produces the desired output. By running a GA multiple times they obtain multiple patterns that map to a particular output.

CHAPTER FOUR
PROPOSED APPROACH

The proposed approach consists of the following phases:

- explicit and implicit data gathering by the HbbTV application,
- automatic detection of number and elements of clusters by CCGA,
- ANN weight initialization by the GA,
- learning by the ANNs, and
- recommending programs again by the HbbTV.

The data gathering part is done by the HbbTV application running on the receiver. The HbbTV application sends explicit and implicit data from multiple users to the server. At the server side, the number and the members of clusters are detected automatically by CCGA, a GA based method proposed in this thesis. This is done using the explicitly collected data. Next, a separate ANN is formed for the users of each cluster. The weights of these ANNs are initialized using GA. Then they are trained and tested again at server by using both explicit and implicit data. At the end, the same HbbTV application fetches the recommendations from the server for each user and shows them on the TV screen.

In this model, tasks are shared between the receiver and the server. Receiver side is only responsible for running the HbbTV application to collect usage data and to present recommendations whereas heavy tasks such as learning and preparing recommendations are performed on server side. The overall system diagram is given in Figure 4.1 showing receiver and server side task sharing.

4.1 Phase 1 – Explicit and Implicit Data Gathering

In order to create user profiles, a data set must be collected. The data are populated while multiple users watching multiple TVs hundreds of hours of TV content using HbbTV catch-up application, such as, football match of last weekend, soap opera of last night, etc. (Figure 4.2).

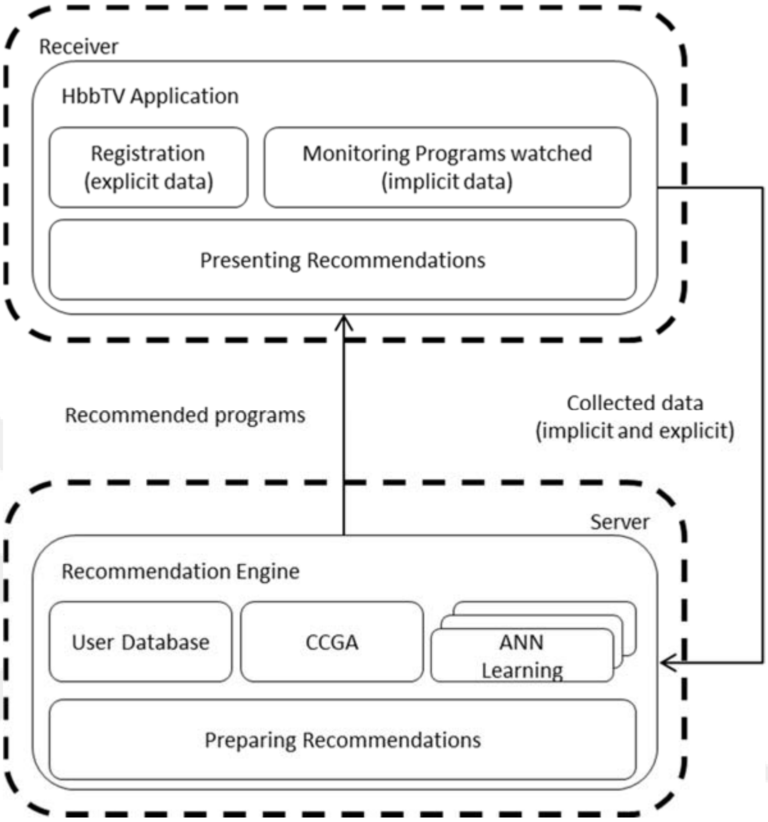

Figure 4.1 Block diagram of the proposed system.

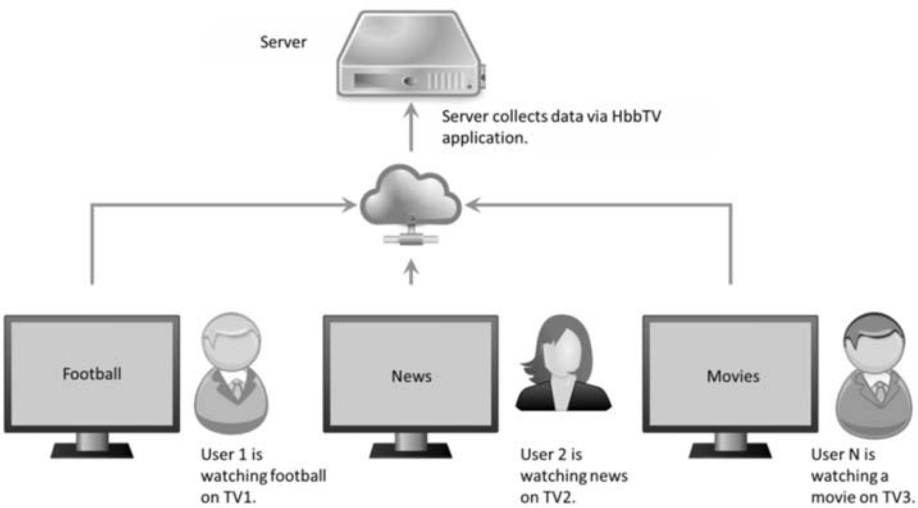

Figure 4.2 Phase 1 - Data collection.

Both explicit and implicit data are collected by an HbbTV application. Explicit data refer to the information entered by the user directly where implicit data are sent to the server without direct interaction of the user.

An HbbTV application is written to prove that data could be collected by this way. Not only the application but the environment to host it (server, etc.) is also setup.

For DVB signaling, a real-time on-air transport stream from the UK (at 666 MHz, channel 45) is recorded. Since there is no HbbTV application on this transport stream, the HbbTV signaling information, in the form of AIT as defined in ETSI specification (ETSI, 2013), is created and injected into this stream (Figure 4.3).

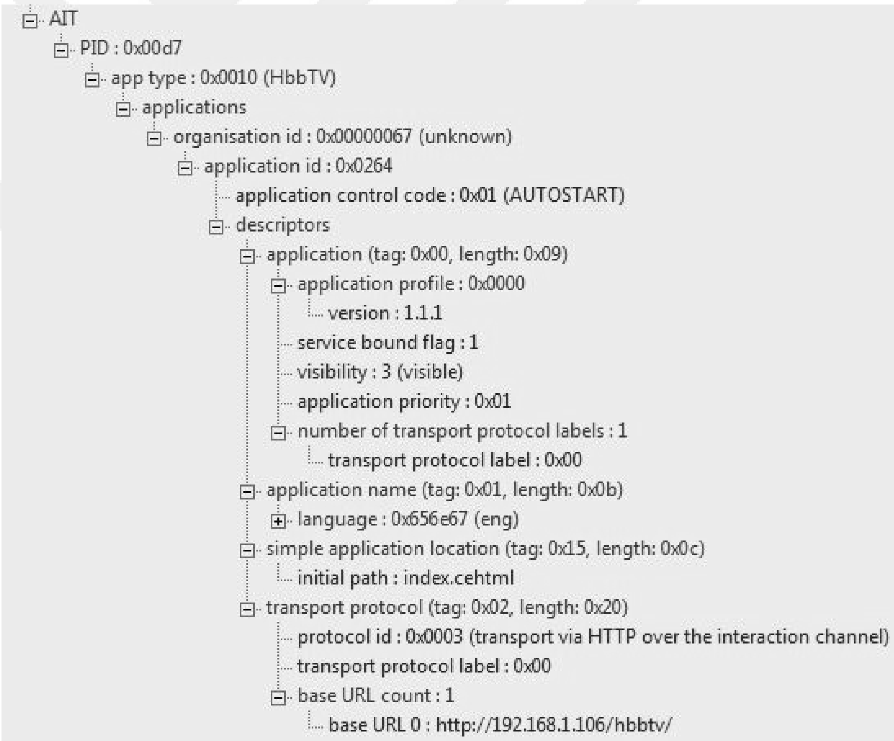

Figure 4.3 AIT injected to the UK broadcast channel at 666 MHz.

Then, the modified stream is played out using a DVB-T modulator. A digital television with HbbTV capability is used for the experiments. No modification is made to the TV software. This means that consumers can benefit this method with all kinds of consumer electronics devices.

4.1.1 Explicit Data Gathering via User Registration

The algorithm of the HbbTV application for the user registration part is as follows:

- The user tunes to a channel.
- Auto-start HbbTV application is launched.
- HbbTV application checks with the server whether this user has already been registered or not.
- If not registered, HbbTV application pops up a window on the TV screen and asks for user’s consent, age, gender, and the top five genre preferences for three different times of the week: weekday evenings, weekend day time, and weekend evenings (Figure 4.4). This explicitly gathered information is sent to the server and stored based on user’s IP address (Figure 4.5).
- This operation is done only once per TV.

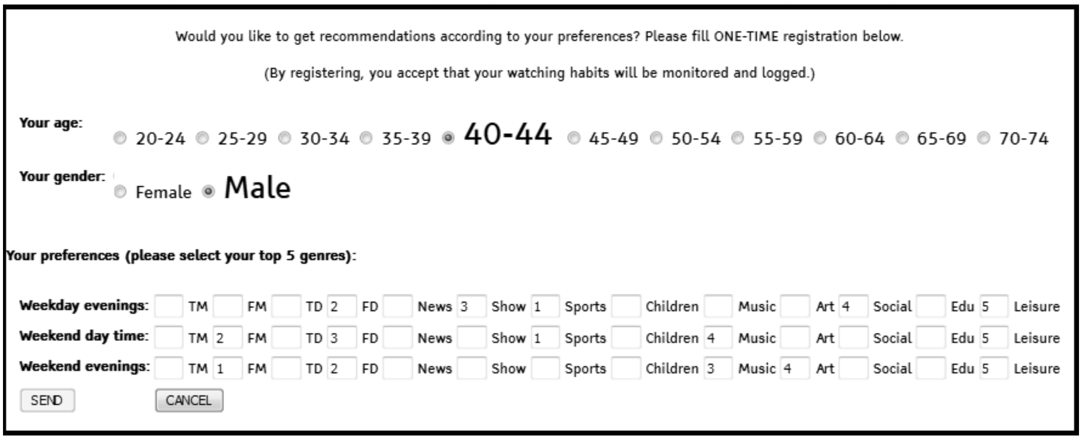

Figure 4.4 Registration form on TV screen.

Getting the user’s consent in advance has vital importance due to privacy matters. If the user does not agree to share her/his data, s/he can close this window by clicking the CANCEL button.

←T→	Nr	IP	Age	Gender	WDE1	WDE2	WDE3	WDE4	WDE5
Edit Copy Delete	13	192.168.1.25	30-34	Male	7	4	9	6	2

WED1	WED2	WED3	WED4	WED5	WEE1	WEE2	WEE3	WEE4	WEE5	Suggestions
7	4	9	2	13	7	4	9	2	13	0

Figure 4.5 Database at the server after user registration.

The list of the genres is taken from the European Standard “Digital Video Broadcasting (DVB); Specification for Service Information (SI) in DVB systems” (ETSI, 2016a), as shown in Table 4.1.

Table 4.1 List of genres

Movie	Show	Children
Drama	Music	Social
Sports	Arts	Education
Leisure	News	

This genre information is carried in the transport stream as 1-byte information in the “Content descriptor”. The first nibble represents the general classification as in Table 4.1 (like “Sports”) where the second nibble is the detailed information (like “Football”). In this study, only the first nibble is taken into consideration.

There is one difference though here; the genres Movie and Drama are extended to hold the difference between Turkish movie (TM) / Turkish drama (TD) and foreign movie (FM) / foreign drama (FD). Since local movies and dramas are very popular in Turkey, it is thought that considering them as separate genres would give better results and would be beneficial to the Turkish consumers.

4.1.2 Implicit Data Gathering While Watching Linear TV

After the registration, the HbbTV application continues to collect data while the user watches linear TV or on demand programs. This is called implicit data gathering since it is done automatically by the system without any user input. The steps are as follows:

- The user tunes to a channel.
- Auto-start HbbTV application is launched with transparent background.
- Since the user has already registered to the server, s/he is not asked again.
- Red button is shown on the bottom right corner to inform the user that HbbTV application exists. The user does not press the red button in this scenario.
- HbbTV application gets the duration of the current event (e.g., Eastenders) on this channel.
- If more than 75 % of the event duration is left, HbbTV application sets a time-out function for the 75 % of the event duration later to check whether the user still watches the same event.
- If the user tunes to another channel within this interval, then the previous HbbTV application gets killed and a new one is launched from the scratch on the new channel.
- If the user stays on the same channel but the remaining of the current event is less than 75 % of the whole duration, then the HbbTV application discards the current event and waits for the next event (Figure 4.6).

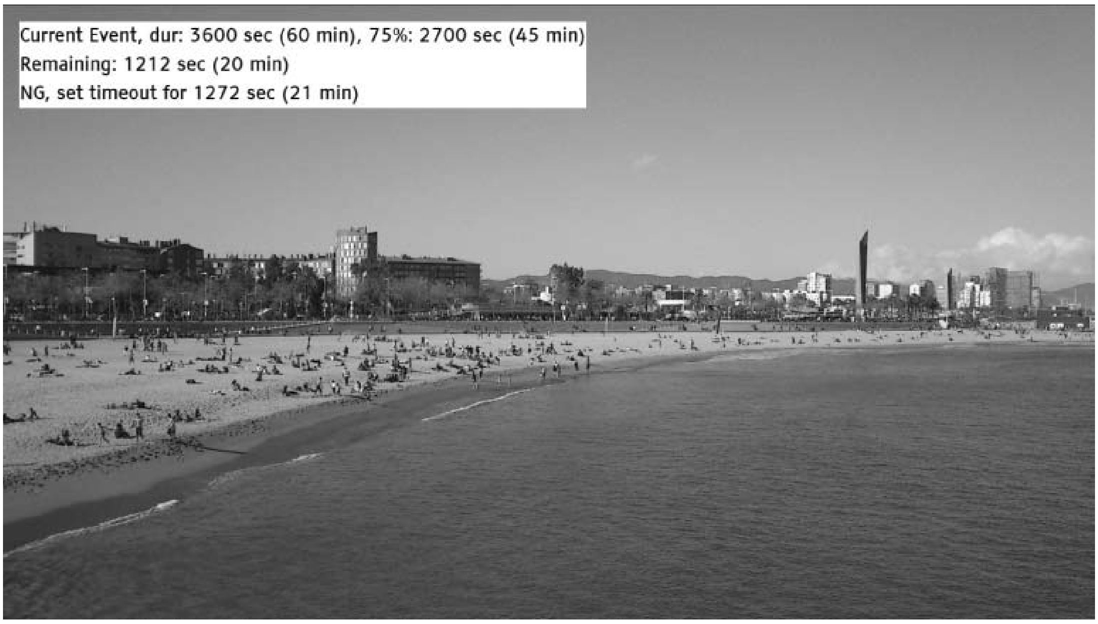

Figure 4.6 HbbTV debug screen - time left is less than 75 % of its duration.

- On the other hand, if the user stays on the same channel during the 75 % of the current event, then that event is accepted as valid data and its details are sent to the server to be added to the database (Figure 4.7).

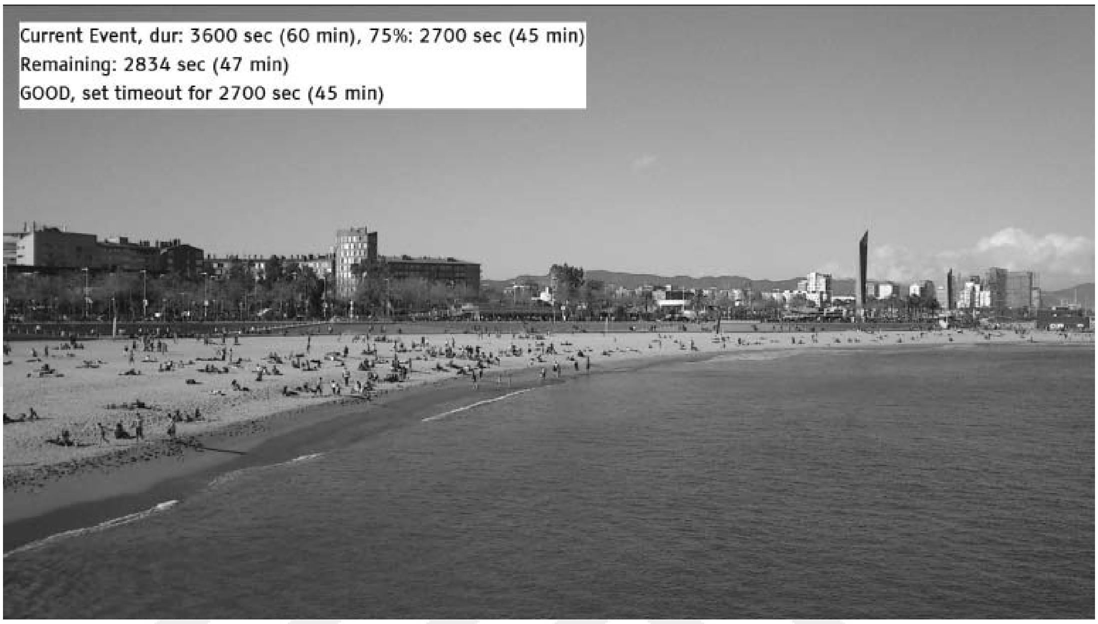

Figure 4.7 HbbTV debug screen - time left is more than 75 % of its duration.

An HbbTV application running on a device supporting HbbTV v1.1.1 can access the name, start time and duration of an event, but it cannot access to the genre information of that event. Therefore, the HbbTV application needs to send event related info to the server so that genre can be found at the server side, which is an extra step to carry (Figure 4.8).

Nr	IP	Age	Gender	TimeOfDay	DayOfWeek	ServiceID	ServiceName	EventID	EventName	EventStartTime	Date
1	192.168.1.92	35-39	Male	20:49	Sat	40976	BBC FOUR HD	dvb://233a.ffffa000.ffffa010;c23c	Bright Lights, Brilliant Minds: A...	Sat, 31 Jan 2015 20:00:00 GMT	2015-01-31 20:49:16.000000

Figure 4.8 Database at the server with valid linear TV data (HbbTV v1.1.1 case).

However, the HbbTV specification version 1.2.1 has an enhancement in this area: It allows applications to access the genre information from the supported device as well as the other parameters. Hence, the HbbTV application sends less data to the server and there is no need to carry an extra step at the server side to find the genre information.

A piece of pseudocode on how to access genre information by an HbbTV application is given in Figure 4.9.

```
<object id="video" type="video/broadcast"></object>
VBObject = document.getElementById('video');
programmes = VBObject.programmes;
currentEvent = programmes.item(0);
name = currentEvent.name;
startTime = currentEvent.startTime;
duration = currentEvent.duration;
descriptors = currentEvent.getSIDescriptors(0x54);
contentDescriptor = descriptors.item(0);
genre = contentDescriptor.charCodeAt(2);
```

Figure 4.9 HbbTV application pseudocode.

Accessing to the DVB-SI descriptors is mandated by the HbbTV specification version 1.2.1, and therefore, it is possible to get the genre information from the content descriptor. This reduces the size of the data to be sent to the server and steps to find the genre information at the server side.

4.1.3 Implicit Data Gathering While Watching Catch-up TV

The HbbTV application is capable of presenting programs as IP-streaming and sending implicitly the information of the selected catch-up content to the server. The algorithm for such a case is as follows:

- The user tunes to a channel.
- Auto-start HbbTV application is launched with transparent background.
- Red button is shown on the bottom right corner to inform the user that HbbTV application exists (Figure 4.10).
- The user presses the red button to launch the catch-up page of the HbbTV application. Available content is shown on the TV screen (Figure 4.11).
- The user selects a program from the catch-up list and starts watching.

- If the user watches the event through the end, then that event is accepted as valid data and the HbbTV application sends its genre to the server to be added to the database (Figure 4.12).

Figure 4.10 Red button image on television screen.

Figure 4.11 Catch-up portal of the HbbTV application.

	Nr	IP	Age	Gender	TimeOfDay	DayOfWeek	Genre	Date
Edit Copy Delete	1	192.168.1.92	35-39	Male	22:31	Tue	Leisure / Hobbies	2015-01-27 22:31:31.000000

Figure 4.12 Database at the server with valid catch-up TV data.

4.2 Phase 2 – CCGA: Controlled Clustering with Genetic Algorithms

A method based on GA for automatic detection of the number and elements of clusters is proposed by Bandyopadhyay and Maulik which is summarized in Section 2.4.2.2. That method is modified in this work to adapt the algorithm to the needs of the problem at hand.

Since the cluster data are used in the ANN learning in this work, it is important to find the optimum number of the ANNs and corresponding I/O sets. Undersized clusters result in large number of ANNs which is inefficient since every new ANN means an additional time and cost. Oversized clusters on the other hand, cause less number of ANNs but longer training times, which is again an unwanted situation.

Keeping the number of elements in the clusters under control and having a distribution of elements into clusters between the pre-defined limits is the ideal case. In order to achieve this, a penalizing transformation is introduced for the fitness function calculation.

First the undersized or oversized clusters are identified by setting limit values n_{min} and n_{max}.

$$n_{\min} \leq n \leq n_{\max} \qquad (4.1)$$

All the clusters having number of elements n between these minimum and maximum values are considered as ideal cases and no penalty is applied for them. Clusters with number of elements less than the minimum number and clusters with number of elements greater than the maximum number are considered as undersized and oversized, respectively; and, undergone to the penalizing transformation.

In the existing method (Bandyopadhyay & Maulik, 2002), the fitness function is inversely proportional to the DB index. Therefore, the smaller DB index is the better fitness value, or the larger DB index is the worse fitness value.

The DB index is a measure of how clusters are scattered and separated. A less scattered, distant cluster has a relatively smaller DB index, yielding to a larger fitness function. A cluster whose elements are spread over a large area and close to the other clusters has a larger DB index and a smaller fitness function.

The idea introduced here is to use a penalizing transformation for undersized or oversized clusters to force a smaller fitness function for them. This can be achieved by increasing their DB index which is equivalent to increasing their scatter value *S*. Then the new scatter value *S′* becomes as below. The graphical representation of the penalizing transformation is given in Figure 4.13.

$$S' = f(S) = \begin{cases} \dfrac{n_{\min}}{2} & n = 1 \\[2ex] \dfrac{n_{\min}^2 + (2 - n_{\min})n - 2}{2(n_{\min} - 1)} S & n < n_{\min} \\[2ex] S & n_{\min} \le n \le n_{\max} \\[2ex] \dfrac{n}{n_{\max}} S & n > n_{\max} \end{cases} \tag{4.2}$$

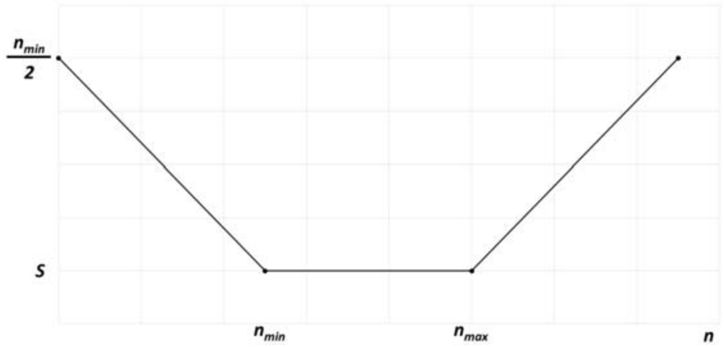

Figure 4.13 Graphical representation of the penalizing transformation.

As seen from the graph, there is no penalty for the clusters whose sizes are between the limits. For outside the limits, a linear penalty is introduced. As cluster size gets further away from these boundaries, the induced penalty increases. Then the modified DB index for the kth chromosome becomes

$$DB'_k = \frac{1}{r_k} \sum_{i=1}^{r_k} \max_{j,\, j \neq i} \frac{S'_i - S'_j}{d_{ij}} \tag{4.3}$$

and the modified fitness function is calculated as

$$F'_k = \frac{1}{DB'_k} \tag{4.4}$$

The DB index of the clusters whose number of elements is beyond these borders is increased; hence the fitness function of those clusters is decreased. This means they have less chance of reproduction. Therefore, over the course of GA iterations, the number of elements *n* in each cluster is limited to a value between a minimum and a maximum figure.

The explicit data (age, gender, and genre preferences) collected in Phase 1 are used to cluster users by CCGA. Content of the chromosomes (genes) comes from the data given explicitly by the user: age, gender, the first 5 preferences of weekday evenings (WDE), the first 5 preferences of weekend day time (WED), the first 5 preferences of weekend evenings (WEE).

Age is represented by a single value, the minimum of the corresponding age group. For example, if user selects 40-44 as his age group, then 40 is used as the chromosome gene. Gender is given as either 0 (female) or 1 (male). Genre preferences are represented by a number between 1 and 13. Genre assignments are given in Table 4.2. Then the gene values are normalized between zero and one.

Table 4.2 Genre values used as genes in the chromosomes

Genre	Value
TM	1
FM	2
TD	3
FD	4
News	5
Show	6
Sports	7
Children	8
Music	9
Art	10
Social	11
Education	12
Leisure	13

A sample chromosome is shown in Figure 4.14. As can be seen here, the dimension (length) of a chromosome becomes 17. The numbers shown are the values before normalization. For reproduction, single-point crossover is used. An example is given in Figure 4.15.

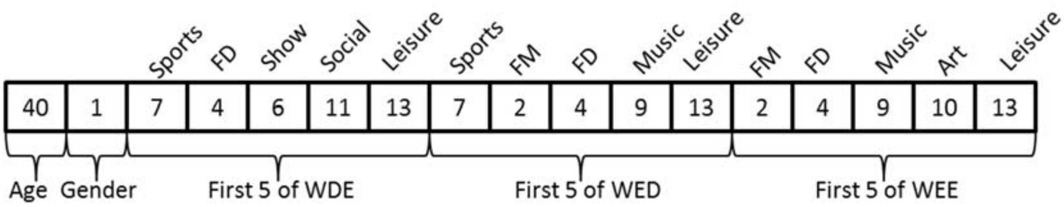

Figure 4.14 A sample chromosome from the population.

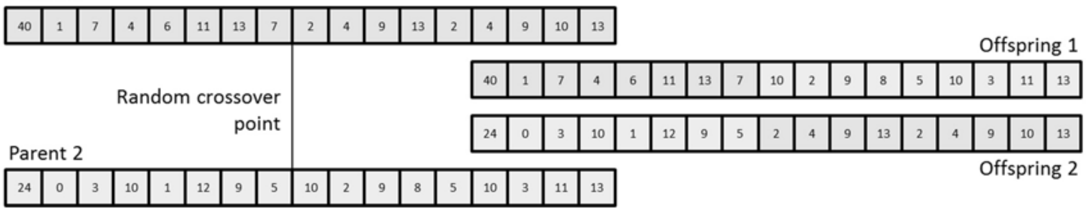

Figure 4.15 Crossing over two parents to produce offsprings.

Mutation probability is set to 1 %. If it happens in age locus, then the mutated age is calculated as:

$$Age' = 70 - Age \tag{4.5}$$

Similarly, the mutated gender and genre are found as:

$$Gender' = 1 - Gender \tag{4.6}$$

$$Genre' = 13 - Genre \tag{4.7}$$

This is shown in Figure 4.16. Above equations and below figure show the values before normalization to give a better idea.

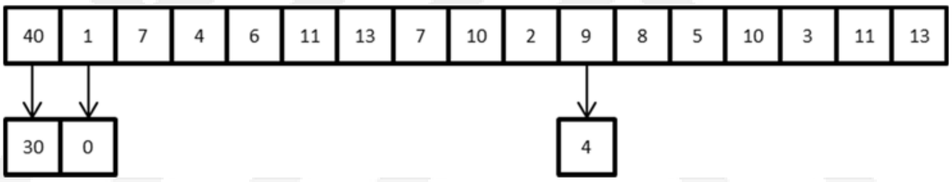

Figure 4.16 Mutated genes.

4.3 Phase 3 – ANN Weight Initialization by GA

In this work, a four-layer MLP model is used as described in Section 2.2. The input layer consists of the variables explained in the next section. The first and the second hidden layers have 30 neurons each, a value selected close to the number of inputs to avoid under-fitting (small learning capacity because of too few neurons) and over-fitting (not being able to train because of too many neurons). The output layer has 13 neurons representing the number of the genres. The connections between these layers are represented by weights and they need to be initialized before the learning starts.

The general trend is to start learning with random weights. However, it is demonstrated that the weight initialization by GA gives good results for supervised learning classification problems (Maulik & Bandyopadhyay, 2000). Therefore, this approach is taken here. A separate ANN is formulated for each cluster constructed in the previous section and their weights are initialized by GA as detailed in Section 2.4.3.

Parents are selected according to Roulette-wheel algorithm. After single-point crossover and mutation with 1 % probability (which is simply changing the sign of the weight value), offsprings are created and replaced by the parents. The resulting weights are stored to be used in the next phase as the ANN initial weights. There is no ANN learning involved in this phase.

4.4 Phase 4 – ANN Learning

ANN learning takes place after the weight initialization explained in the previous section. Constructed ANNs are trained with 90 % of the data collected by the HbbTV application for the respective cluster members in Phase 1. The 10 % is allocated for testing only and never given to the networks during the training. At each epoch, training data are presented in random order and weights are updated after each input/output sample. At the end of each epoch, test data are given and error is calculated but weights are not updated. If the average test error is less than the average test error of the previous epoch, then the new error becomes the minimum test error and weights are stored. Learning is terminated either at a pre-determined epoch or at a test error below a pre-determined value and stored weights are used for the program recommendation phase.

In the equations given in Section 2.2, η is taken as 0.5, γ and α as 0.9 where all of them are typical values. They are chosen as “balanced” figures, which cause neither overshoot nor slow convergence. Learning starts with the weights initialized by GA in the previous phase.

4.4.1 ANN Input Selection

The input layer of the ANNs consists of user information, the hour of the day, the day of the hour, gender, age and available genres of the time of concern. The detailed information for each input parameter can be found in the next sections.

4.4.1.1 User Inputs

It is obvious that everybody has different taste of programs and preferences. People having the same gender and the same age can have different viewing habits. Therefore, users should be differentiated and represented in the input layer uniquely. In this work, binary representation is selected.

4.4.1.2 Hour Input

It is expected that the time of the day should play an important role and the programs watched in different hours have different genres. Therefore, it is meaningful to use hour information as an input parameter for this model.

It is decided to take the starting hour of the programs. In order to make sure all inputs having the same range, the hour values are normalized into the interval [0, 1] by dividing each value by 24.

4.4.1.3 Day Input

Similar to the hour of the day, the day of the week should also be an important parameter. Especially drama programs have weekly cycles; therefore, people watch programs of different genres every day. Thinking that it would be helpful in ANN learning, the day information is taken as an input for this model by being normalized into [0, 1] range.

4.4.1.4 Gender Input

Males and females are expected to have different preferences over the genres of programs they watch. Therefore gender, normalized into [0, 1], is also selected as an input variable.

4.4.1.5 Age Inputs

It is expected to see similar viewing habits in closer age groups. Hence, it is necessary to reflect that the adjacent age groups should have closer input values than the distant age groups. Therefore it is decided to express the age with 11 shifted sinusoidal inputs, as shown in Figure 4.17, representing 11 age groups. By this way, the age inputs of closer age groups will take closer values.

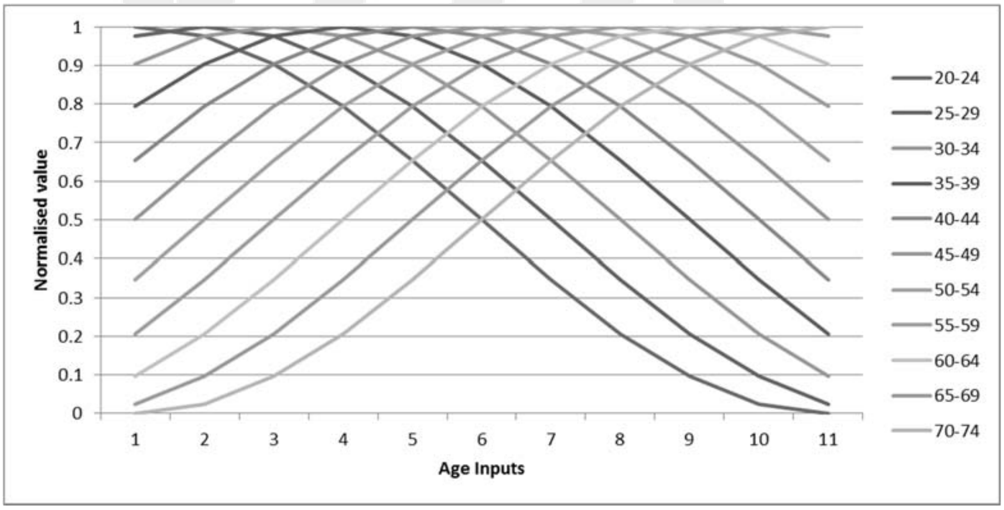

Figure 4.17 Age input values according to the age intervals.

4.4.1.6 Genre Inputs

Television viewing preferences are based on the availability of the programs from a collection of different genres at the time of watching. Sometimes users have to choose certain genres where their first, second, or third preferences are not available. This means that the ANN model should know which genres were available when a certain genre was chosen by the user.

Therefore, of all 13 genres, one of them being the output, the other 12 are given as inputs, having value 0 or 1, representing their availability at the time of watching.

4.4.2 ANN Output Selection

The output layer of the ANNs consists of 13 neurons where each of them represents a program genre. The one with the greatest value is considered as the output genre while generating the recommendations.

During the training and testing, the desired values for these neurons are assigned as “1” for the genre the user has watched and “0” for the others.

4.4.3 Incremental Learning Method

In real life, data collection by the HbbTV application is not an instantaneous action, rather it is a process. Some users can watch ten programs of the same genre in a less time than some other users. Thus, it is not necessary to wait for all users to reach ten data to start learning. It can be an incremental algorithm.

In this model, ANN learning immediately starts when at least one user has watched at least ten programs from a genre. Other users and genres are gradually included to the training and testing sets when they have reached at least ten data. Learning is performed in daily basis (e.g., every night at 03:00 am) so that the new users of that day can be included in the system. ANN weights of minimum test error are saved for the next day’s training.

4.5 Phase 5 – Program Recommendation

The ANN model successfully constructed after training and testing phases is used to make recommendations according to user’s viewing habits. Recommendations are provided to the user via the same HbbTV application which communicates with the “Recommendation Engine” at the server side (Figure 4.18). The recommendation engine uses the trained and tested ANN of the cluster where the user belongs and calculates the output vector by providing the necessary inputs and using the stored weights. The greatest value of the output vector is considered as the ANN output and the genre associated with that output neuron is taken as the genre to be recommended. Then, the recommendation engine lists the current programs where their genre matches with the ANN output and the HbbTV application fetches this list to present to the user.

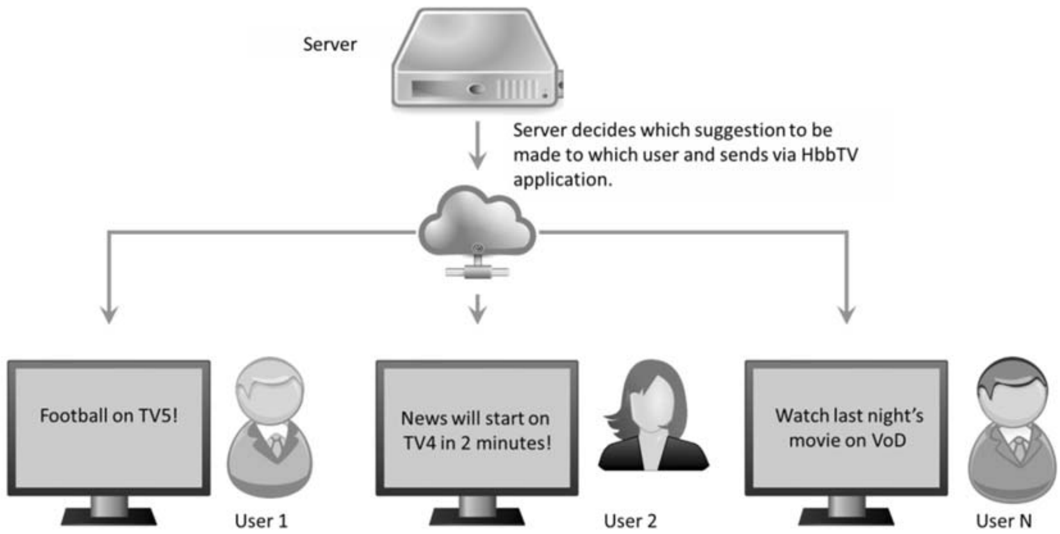

Figure 4.18 Phase 5 - Program recommendation.

Since the ANN waits for the user to watch at least ten television programs of the same genre to start learning, a gap occurs between the user registration and the first time when the ANN produces recommendations. In order to fill this gap, the recommendation engine makes use of the explicit data given during the registration. It prepares the programs matching user's preferences specified on the registration screen. Then, the HbbTV application presents these programs as recommendations. This auxiliary process goes on until the recommendations based on the real watching data as ANN output become ready.

The algorithm of the Program Recommendation HbbTV application is as follows:

- The user turns on the TV and tunes to a channel.

- Auto-start HbbTV application is launched with transparent background.
- HbbTV application checks with the server whether there are any program recommendations for this particular user for the time of concern (Figure 4.19).

←T→	Nr	IP	Age	Gender	WDE1	WDE2	WDE3	WDE4	WDE5
Edit Copy Delete	1	192.168.1.25	40-44	Male	7	4	6	11	13

WED1	WED2	WED3	WED4	WED5	WEE1	WEE2	WEE3	WEE4	WEE5	Suggestions
7	2	4	9	13	2	4	9	10	13	1

Figure 4.19 Database at the server when recommendations are available.

- If so, HbbTV application fetches the list of contents suggested by the recommendation engine at the server. The recommendation engine either utilizes user's explicit data to construct recommendations or provides the necessary inputs for the ANN for producing the genre outputs at the server side.
- The HbbTV application shows this list of recommendations on the TV screen (Figure 4.20).

Figure 4.20 Recommendations by the HbbTV application on the TV screen.

- The user either makes a selection and the HbbTV application tunes the TV to that channel or closes the application.

- If the user does not make any selection, the HbbTV application removes the list from the TV screen after some time.

4.6 Overall Algorithm

The overall algorithm of the proposed method can be divided into two parts:

- The algorithm running on the receiver side,
- The algorithm running on the server side.

The following figures demonstrate how these algorithms work. Figure 4.21 is the flowchart of the HbbTV application, which is launched when the user tunes to a channel.

Data gathering and program recommendation are continuous processes (Figure 4.21). However, CCGA and ANN learning are repeated at the server side periodically. Flowchart of this periodic algorithm can be drawn as in Figure 4.22 which shows the daily routine.

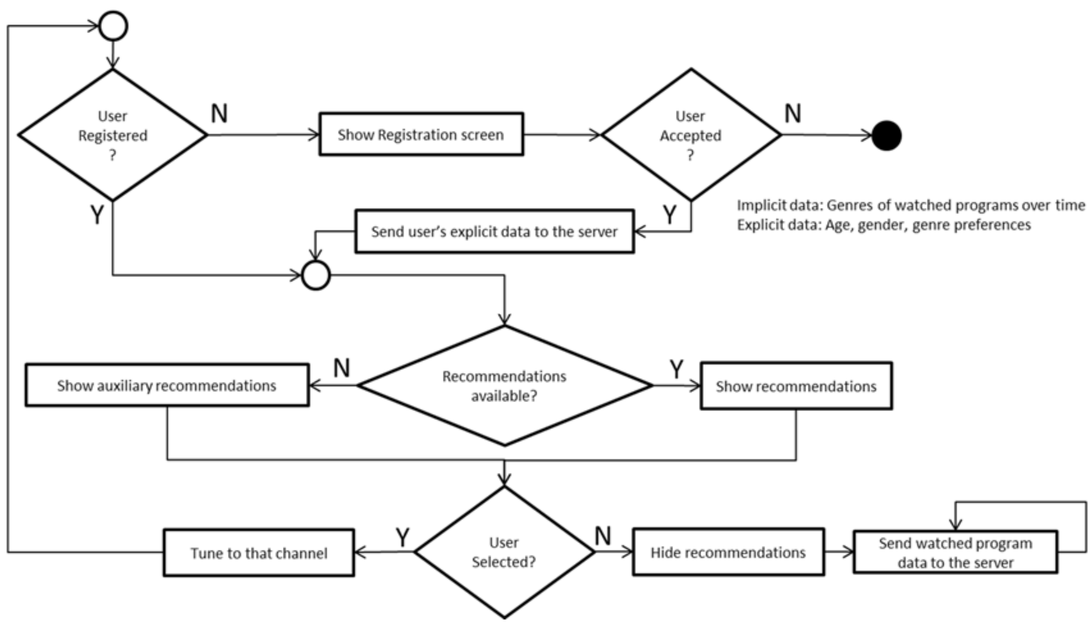

Figure 4.21 HbbTV application flowchart at receiver side.

As explained in Section 4.4.3, ANN learning is repeated every night. CCGA frequency however is lower than that (every week, etc.) under normal circumstances. However, if the number of the users of an ANN reaches the upper limit (e.g., 100, a pre-defined number depending on the server computation power), CCGA is performed extraordinarily. In such a case, ANN learning follows CCGA, even if it is not the dedicated time.

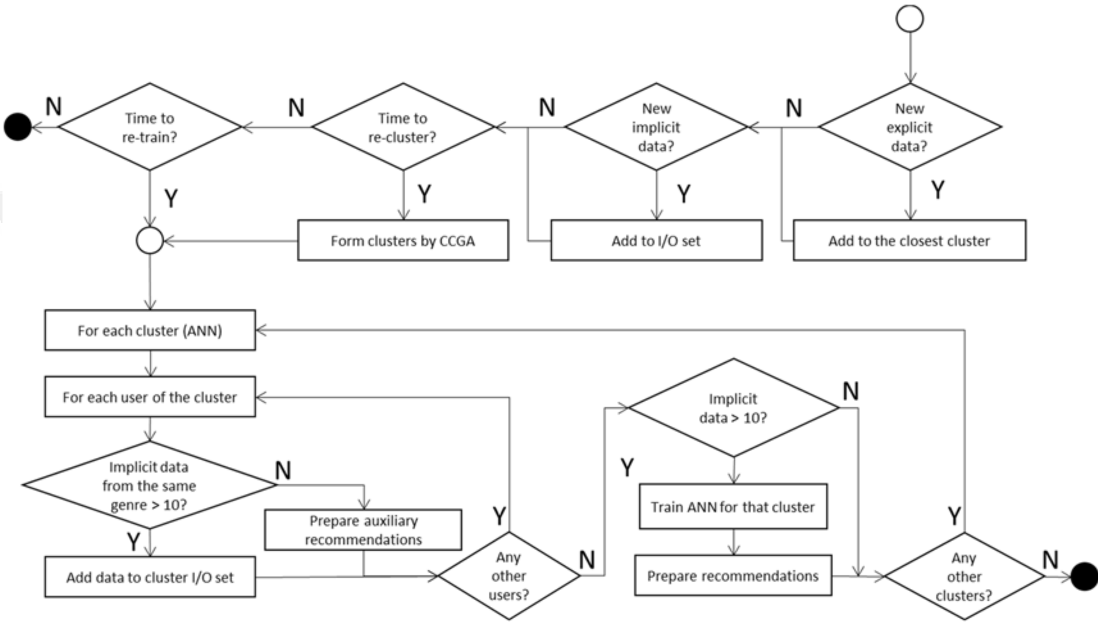

Figure 4.22 Flowchart of the periodic server operation.

CHAPTER FIVE
EXPERIMENTS AND RESULTS

In this chapter, the methods proposed in the previous chapter are implemented and validated by data collected in several ways. Clustering and ANN learning examples are performed and results are presented. GA clustering is compared with K-means clustering with pre-defined number of clusters. The method of Bandyopadhyay and Maulik (2002) is also applied. Experiments on CCGA to find number and elements of clusters automatically are performed; results of random weight initialization are compared with GA-based weight initialization. The details can be found in the following sections.

5.1 Data Collection

Although it is possible to gather data by an HbbTV application as described in the previous chapter, it has practical difficulties. HbbTV information should be added to the real DVB signal of the broadcaster, which is beyond the focus of this thesis.

Therefore, two other approaches are taken to collect the data as if it came from the HbbTV application. The first one is data collection by an on-line questionnaire where participants provide age, gender and genre preferences. These data are used as the explicit information. The second data collection method is manually recording the programs watched. The data gathered by this method are used as the implicit information.

5.1.1 Data Collection by Questionnaire

In order to gather data from multiple users, a questionnaire is prepared and put on a Web site. Then, it is announced through social media to the people and kindly asked them to fill their preferences according to the genres.

The questionnaire is prepared both in English and Turkish in order to reach more people. Before starting, it is possible to choose the language (Figure 5.1).

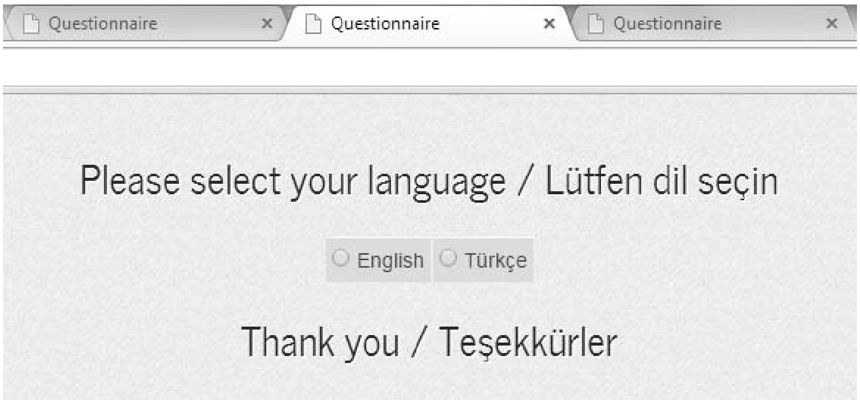

Figure 5.1 Language selection for the questionnaire.

On the questionnaire page, the audience is asked to enter which TV programs they would watch in the evening of a weekday, in the daytime of the weekend, and in the evening of the weekend. Their first five preferences described below are requested.

1: The 1st choice when programs of all genres are available.
2: The 2nd choice when no program of the 1st choice is available.
3: The 3rd choice when no program of the 1st and 2nd choices is available.
4: The 4th choice when no program of the 1st, 2nd, and 3rd choices is available.
5: The 5th choice when no program of the 1st, 2nd, 3rd, and 4th choices is available.

An example on how to fill the table is also provided on the Webpage (Figure 5.2). Together with the program preferences, participants' age range and gender are also asked as mandatory information (Figure 5.3). Name was optional.

In total, 248 people responded to the questionnaire. The number of male participants (147 people, 59 %) are slightly higher than the number of females (101 people, 41 %), as shown in Table 5.1 and Figure 5.4.

Example:

Genre	Weekday evening (Mon – Fri) between 19:00 – 24:00	Weekend day time (Sat – Sun) between 13:00 – 19:00	Weekend evening (Sat – Sun) between 19:00 – 24:00
Turkish Movie (general, detective/thriller, adventure/western/war, science fiction/fantasy/horror, comedy, soap/melodrama/folkloric, romance, serious/classical/religious/historical, adult, etc.)			2
Foreign Movie (general, detective/thriller, adventure/western/war, science fiction/fantasy/horror, comedy, soap/melodrama/folkloric, romance, serious/classical/religious/historical, adult, etc.)	3	1	
Turkish Drama (general, detective/thriller, adventure/western/war, science fiction/fantasy/horror, comedy, soap/melodrama/folkloric, romance, serious/classical/religious/historical, adult, etc.)	1	4	1
Foreign Drama (general, detective/thriller, adventure/western/war, science fiction/fantasy/horror, comedy, soap/melodrama/folkloric, romance, serious/classical/religious/historical, adult, etc.)			
News / Current affairs (general, news/weather report, news magazine, documentary, discussion/interview/debate, etc.)			
Show / Game show (general, game show/quiz/contest, variety show, talk show, etc.)			
Sports (general, special events (Olympic Games, World Cup, etc.), sports magazines, football/soccer, tennis/squash, team sports (excluding football), athletics, motor sport, water sport, winter sports, equestrian, martial sports, etc.)			
Children's / Youth Programmes (general, pre-school children's programmes, entertainment programmes for 6 to14, entertainment programmes for 10 to 16, informational/educational/school programmes, cartoons/puppets, etc.)			
Music / Ballet / Dance (general, rock/pop, serious music/classical music, folk/traditional music, jazz, musical/opera, ballet, etc.)	4	5	5
Arts / Culture (without music) (general, performing arts, fine arts, religion, popular culture/traditional arts, literature, film/cinema, experimental film/video, broadcasting/press, new media, arts/culture magazines, fashion, etc.)	5		4
Social / Political issues / Economics (general, magazines/reports/documentary, economics/social advisory, remarkable people, etc.)			
Education / Science / Factual topics (general, nature/animals/environment, technology/natural sciences, medicine/physiology/psychology, foreign countries/expeditions, social/spiritual sciences, further education, languages, etc.)	2	3	3
Leisure / Hobbies (general, tourism/travel, handicraft, motoring, fitness and health, cooking, advertisement/shopping, gardening, etc.)		2	

Figure 5.2 Questionnaire form.

Name:		Age:	20-24 25-29 30-34 35-39 40-44 45-49 50-54 55-59 60-64 65-69 70-74	Gender:	Female Male

Figure 5.3 Required user information on the questionnaire form.

Table 5.1 Gender distribution of people responding to the questionnaire

Gender	Number	Percentage (%)
Male	147	59
Female	101	41
Total	248	100

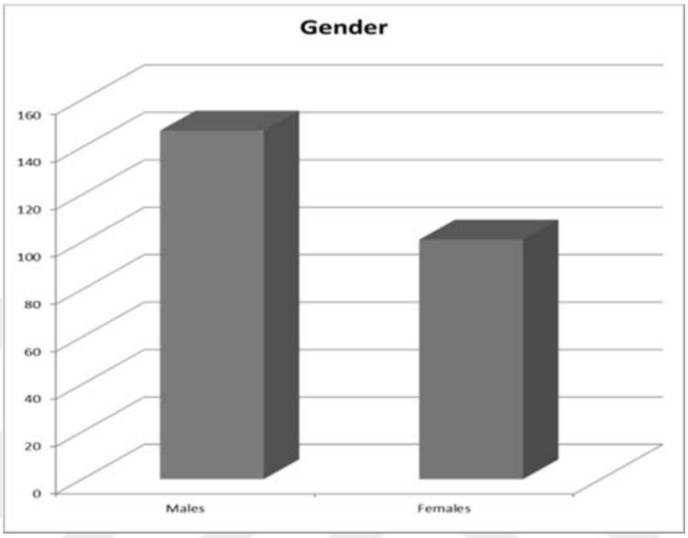

Figure 5.4 Gender ratio of the questionnaire participants.

The age distribution is given in Table 5.2 and Figure 5.5, the range between 20-24 being the highest among the total participants.

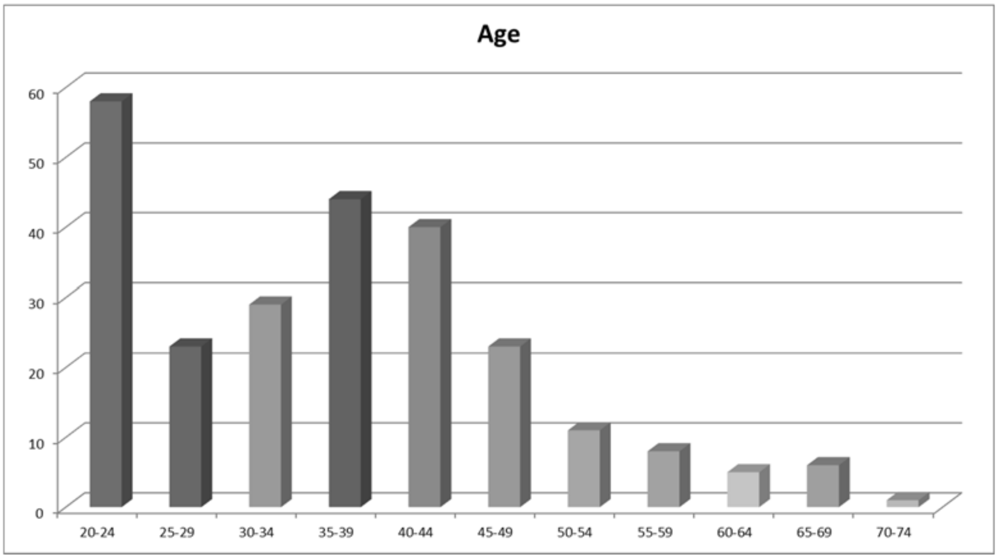

Figure 5.5 The age distribution of the questionnaire participants.

As shown in Table 5.3, among male contributors, the greatest input comes from the age interval 20-24 with 24 %; whereas females aged between 40-44 give the greatest contribution.

Table 5.2 Age distribution of people responding to the questionnaire

Age	Number	Percentage (%)
20-24	58	23.39
25-29	23	9.27
30-34	29	11.69
35-39	44	17.74
40-44	40	16.13
45-49	23	9.27
50-54	11	4.44
55-59	8	3.23
60-64	5	2.02
65-69	6	2.42
70-74	1	0.40
Total	248	100.00

Table 5.3 Age vs gender distribution

	Male		Female	
Age	Number	Percentage	Number	Percentage
20-24	36	24	22	22
25-29	16	11	7	7
30-34	18	1	11	11
35-39	30	20	14	14
40-44	16	11	24	24
45-49	16	11	7	7
50-54	7	5	4	4
55-59	3	2	5	5
60-64	2	1	3	3
65-69	2	1	4	4
70-74	1	1	0	0

"News" genre is the first choice in weekday evenings (Table 5.4). The most preferred genre in weekend daytime and weekend evening is "FM". "FM" is also the weighted first choice, while "Children" genre is the least preferred one in overall.

Table 5.4 Overall analysis of questionnaire responses

Genre	Weighted Sum	1st Choice (WDE)	1st Choice (WED)	1st Choice (WEE)
TM	540	5	2	5
FM	1,799	41	43	95
TD	653	25	6	11
FD	1,265	39	29	24
News	1,381	71	33	36
Show	828	11	7	13
Sports	942	18	36	34
Children	108	2	7	1
Music	538	3	12	7
Arts	468	3	9	3
Social	565	6	8	6
Education	1,056	17	21	9
Leisure	1,017	7	35	4
Total	11,160	248	248	248

Table 5.5 shows that "FM" is the most popular genre among young people, and "News" gets the middle-old aged people's attention. There is a high correlation between closer age groups' genre preferences (Table 5.6 and Figure 5.6).

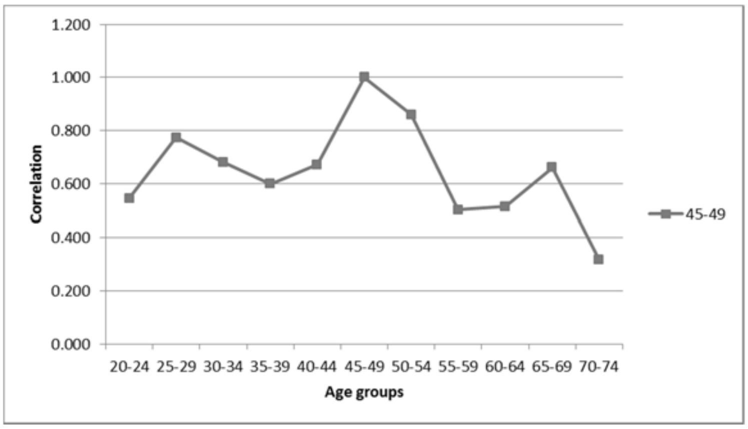

Figure 5.6 Correlation of age group 45-49 with others.

Table 5.5 Genre preferences among different age groups

Age	1st Pref (%)	2nd Pref (%)	3rd Pref (%)	4th Pref (%)	5th Pref (%)
20-24	FM (18)	FD (13)	Sports (9)	Show (9)	News (9)
25-29	FM (15)	News (12)	FD (11)	Sports (11)	Education (10)
30-34	FM (17)	FD (12)	Education (11)	News (10)	Show (9)
35-39	FM (18)	FD (14)	Education (10)	News (10)	Leisure (9)
40-44	FM (17)	FD (14)	News (13)	Leisure (11)	Education (9)
45-49	News (20)	FM (15)	Sports (12)	Education (12)	Leisure (9)
50-54	News (19)	FM (14)	Social (9)	FD (8)	Sports (8)
55-59	News (16)	Leisure (14)	Show (14)	TD (12)	Music (10)
60-64	Music (17)	News (15)	Education (15)	Leisure (13)	Social (9)
65-69	News (23)	TD (13)	Music (13)	Education (10)	Leisure (9)
70-74	News (31)	Music (24)	TM (13)	TD (11)	Social (9)

Table 5.6 Correlation between age groups with respect to their genre preferences

Age	20-24	25-29	30-34	35-39	40-44	45-49	50-54	55-59	60-64	65-69	70-74
20-24	1.000	0.890	0.932	0.941	0.802	0.546	0.550	0.331	0.111	0.141	0.183
25-29		1.000	0.933	0.901	0.842	0.774	0.734	0.311	0.129	0.295	0.120
30-34			1.000	0.946	0.877	0.681	0.688	0.294	0.108	0.269	0.133
35-39				1.000	0.926	0.600	0.612	0.189	0.054	0.122	0.233
40-44					1.000	0.672	0.787	0.210	0.140	0.299	0.052
45-49						1.000	0.860	0.503	0.515	0.662	0.316
50-54							1.000	0.552	0.444	0.662	0.331
55-59								1.000	0.469	0.674	0.542
60-64									1.000	0.791	0.585
65-69										1.000	0.766
70-74											1.000

Figure 5.7 displays the information that the genre preferences differ for each time slot.

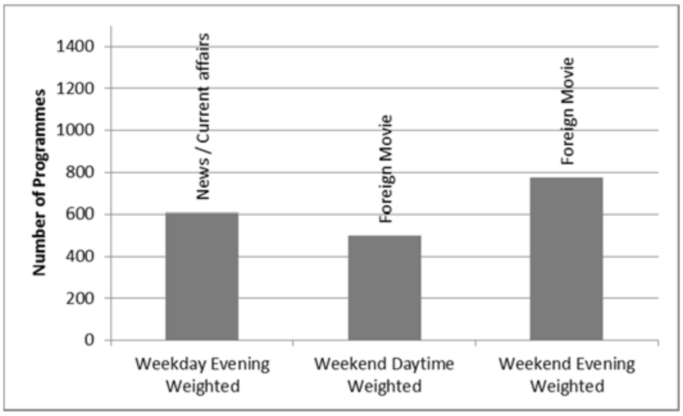

Figure 5.7 Weighted sum of the most preferred genres at each time slot.

Table 5.7 and Figure 5.8 show the gender impact on participants' genre preferences. The greatest differences between males and females are at "Turkish Drama", in favor of females and "Sports", in favor of males.

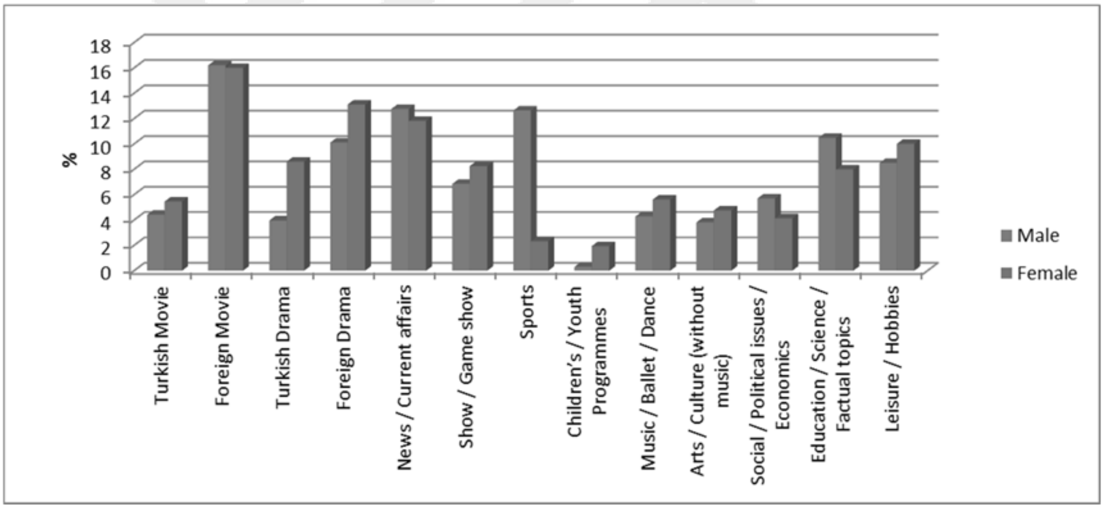

Figure 5.8 Genre preferences of questionnaire participants according to gender.

5.1.1.1 Data Generation out of the Questionnaire

As explained above, the answers to the questionnaire provide five input-output data per person per time (weekday evening, weekend daytime, and weekend evening). Obviously, it is not enough to train the model successfully. Therefore, it is decided to generate more data using the questionnaire responses for each person participated.

Table 5.7 Genre preferences of questionnaire participants according to gender

Genres	Male (%)	Female (%)	Difference (%)
TM	4.41	5.46	-1.04
FM	16.21	16.00	0.21
TD	3.96	8.60	-4.64
FD	10.11	13.11	-3.00
News	12.76	11.82	0.94
Show	6.85	8.25	-1.40
Sports	12.64	2.33	10.31
Children	0.30	1.94	-1.63
Music	4.28	5.61	-1.33
Arts	3.81	4.75	-0.94
Social	5.70	4.14	1.56
Education	10.48	7.99	2.49
Leisure	8.50	10.01	-1.52
Total	100.00	100.00	0.00

The first choice of the user when all 13 genres are available would also mean that the user would make the same choice for all combinations of availability of 12 genres, provided that the first choice genre is always present. This yields $2^{12} = 4096$ alternatives. One of them is already given in the questionnaire, and the other 4095 data are generated by a script.

Similarly, the second choice of the user when her/his first choice is not available but other 12 genres are available would also mean that the user would make the same choice for all combinations of availability of 11 genres, provided that the first choice is not available and the second choice is present. This yields $2^{11} = 2048$ alternatives, 2047 of which are generated by a script.

Repeating the same procedure for the third, fourth, and the fifth choices, 7931 data are prepared per person without breaking the logic. Including participant’s 5 choices, the total number of data per person per time slot becomes 7936.

5.1.2 Data Collection by Recording the Programs Watched

In order to check the model with real TV watching data, 8 people from family members and friends who also responded the questionnaire were asked to fill a spreadsheet (Table 5.8) for an entire month. The responses are not exactly for the same days. The earliest starts from December 27th, 2014 and the latest ends on January 31st, 2015.

Table 5.8 One-week data of one-month real TV watching

Date	Hour	Genre	Hour	Genre	Hour	Genre	Hour	Genre
04/01/15	16:00	Sports	18:00	Leisure	20:00	Education	22:00	Music
05/01/15	20:00	Show	21:00	FD	22:00	News		
06/01/15	19:45	Sports	22:00	FD				
07/01/15	19:45	Sports	22:00	News	22:30	Social		
08/01/15	18:00	Education	20:00	Sports	22:00	FD		
09/01/15	18:00	News	19:00	Leisure	21:00	TM		
10/01/15	14:00	FM	15:00	Sports	17:30	Music	21:00	FD

Out of 8 participants, six of them (75 %) are female and two of them (25 %) were male (Table 5.9 and Figure 5.9).

Table 5.9 Gender distribution of participants

Gender	Number	Percentage (%)
Male	2	25
Female	6	75
Total	8	100

The age distribution is shown in Table 5.10 and Figure 5.10, the range between 65-69 being the highest among the total participants who recorded the programs they watched.

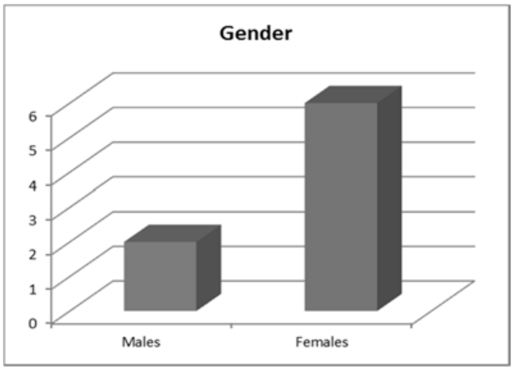

Figure 5.9 Gender ratio of the TV watching participants.

Table 5.10 Age distribution of participants

Age	Number	Percentage (%)
20-24	0	0.00
25-29	0	0.00
30-34	0	0.00
35-39	1	12.50
40-44	0	0.00
45-49	1	12.50
50-54	1	12.50
55-59	0	0.00
60-64	2	25.00
65-69	3	37.50
70-74	0	0.00
Total	8	100.00

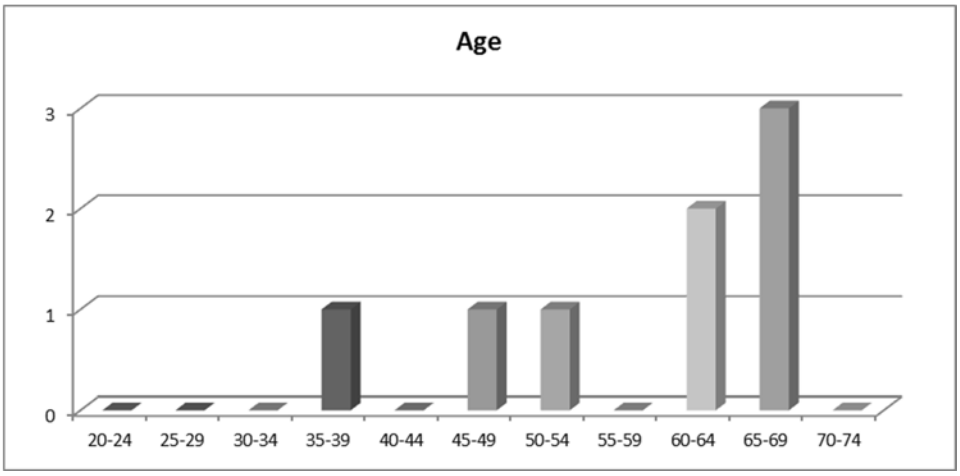

Figure 5.10 The age distribution of the TV watching participants.

Preferences of each participant against genres are figuratively shown below (Figure 5.11). Some participants have narrower interests (for example, User 6 has watched programs from only 4 genres) than the others (for example, User 4 has watched programs from 8 genres).

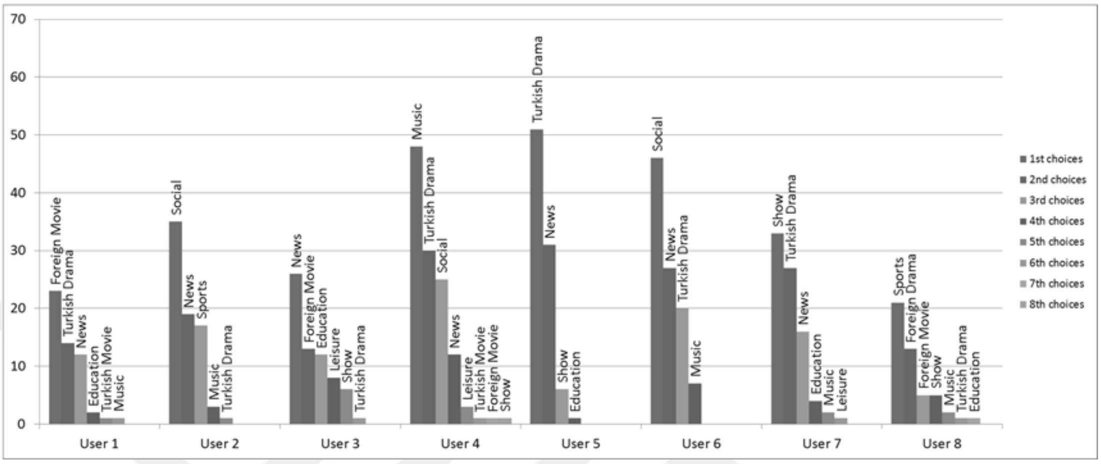

Figure 5.11 Genre preferences of each participant – recorded programs.

Table 5.11 shows the total number of genres each participant recorded. It is seen that User 4 has the greatest input with 121 programs watched, whereas User 8 is the one who watched the least programs (48). In overall, “Turkish Drama” genre is the most watched one. “Children” and “Arts” have no records.

It is seen from user preferences (Figure 5.12) that the time of the day plays an important role and the programs watched in different hours have different genres.

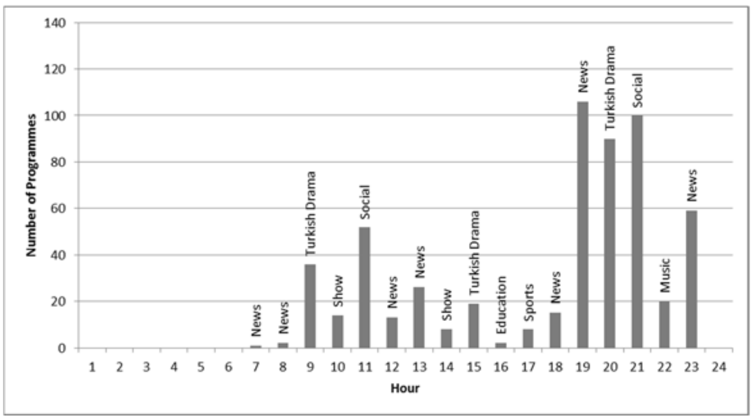

Figure 5.12 Total number of programs watched and hourly most preferred genres.

Table 5.11 Number of programs watched from each genre

Genre	User1	User2	User3	User4	User5	User6	User7	User8	Total
TM	1	0	0	1	0	0	0	0	2
FM	23	0	13	1	0	0	0	5	42
TD	14	1	1	30	51	20	27	1	145
FD	0	0	0	0	0	0	0	13	13
News	12	19	26	12	31	27	16	0	143
Show	0	0	6	1	6	0	33	5	51
Sports	0	17	0	0	0	0	0	21	38
Children	0	0	0	0	0	0	0	0	0
Music	1	3	0	48	0	7	2	2	63
Arts	0	0	0	0	0	0	0	0	0
Social	0	35	0	25	0	46	0	0	106
Education	2	0	12	0	1	0	4	1	20
Leisure	0	0	8	3	0	0	1	0	12
Total	53	75	66	121	89	100	83	48	635

Figure 5.13 shows that people watch programs of different genres every day.

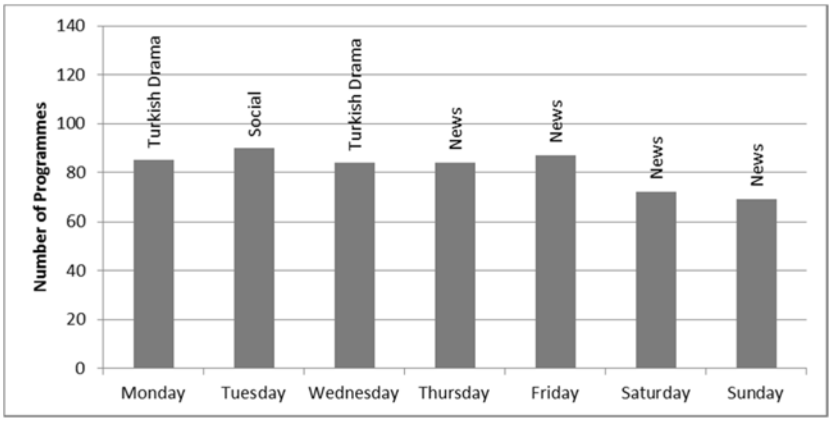

Figure 5.13 Total number of programs watched and daily most preferred genres.

Table 5.12 and Figure 5.14 show the gender impact on participants' genre preferences. As similar to the questionnaire data, the greatest differences between males and females here are also at "Turkish Drama" and "Sports", respectively.

Table 5.12 Genre preferences of recording responses according to gender

Genre	Male (%)	Female (%)	Difference (%)
TM	0.00	0.39	-0.39
FM	4.07	7.23	-3.16
TD	1.63	27.93	-26.30
FD	10.57	0.00	10.57
News	15.45	24.22	-8.77
Show	4.07	8.98	-4.92
Sports	30.89	0.00	30.89
Children	0.00	0.00	0.00
Music	4.07	11.33	-7.26
Arts	0.00	0.00	0.00
Social	28.46	13.87	14.59
Education	0.81	3.71	-2.90
Leisure	0.00	2.34	-2.34
Total	100.00	100.00	0.00

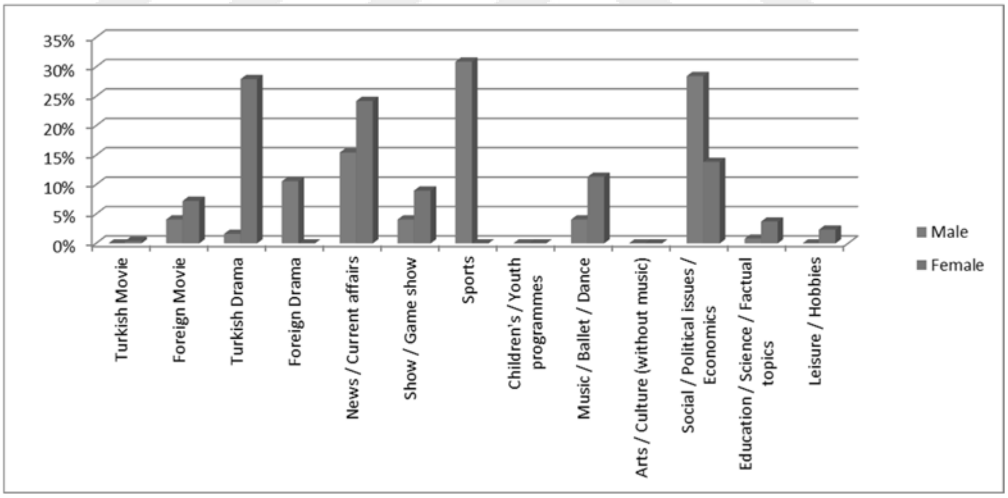

Figure 5.14 Genre preferences of recording responses according to gender.

5.2 Recommendation Results with Questionnaire Data

As explained previously, with the on-line questionnaire method, data from 248 people are gathered. These data are used to simulate the status of the proposed model where 248 people have been registered to the recommendation system, and they have watched TV programs according to their genre preferences.

With these data, six main experiments are conducted:

- Experiment I: Training a pre-defined number (eight) of ANNs by randomly distributing 248 people to these ANNs,
- Experiment II: Clustering 248 people to a pre-defined number (eight) of sets by GA, and then training eight ANNs with these clustered sets,
- Experiment III: Clustering 248 people to a pre-defined number (eight) of sets by K-means algorithm, and then training eight ANNs with these clustered sets,
    - Comparing GA clustering with K-means clustering,
- Experiment IV: Applying the method of Bandyopadhyay and Maulik (2002) to find the clusters automatically,
- Experiment V: Finding the number and members of clusters automatically and clustering 248 people to that number (17) of sets by CCGA, and then training 17 ANNs with these clustered sets with random initial weights,
- Experiment VI: Finding the number and members of clusters automatically and clustering 248 people to that number (17) of sets by CCGA, and then training 17 ANNs with these clustered sets with GA-based initial weights.

For each experiment above, the recommendations are calculated using ANN outputs as defined in Section 4.4.2 and are compared with the user data. Recommendation errors given below are calculated as the percentage of the number of incorrect recommendations with respect to the total number of recommendations. The following sections give the details and results for each of them.

5.2.1 Experiments with Pre-defined Number of Clusters

This section covers three experiments (Experiment I, Experiment II and Experiment III) listed above where the number of clusters is chosen as a static pre-determined value.

5.2.1.1 *Experiment I: Random User Selection*

As the first experiment, the participants of the questionnaire are distributed to eight sets. This distribution was random and each set has 32 users, except the last one which has the remaining 24 users. Then a separate ANN is formed for each of them. Learning of the ANNs is performed using the data of respective users.

In the model based on the questionnaire responses, there is no hour and day granularity, but there are three time zones of the week. Therefore they are represented as 2-bit inputs. The trained ANNs are tested to give recommendations as their output and the results are given in Table 5.13.

Table 5.13 Recommendation results for random user sets (Experiment I)

ANN	User Set	Recommendation Error (%)
1	1 – 32	1.05
2	33 – 64	1.58
3	65 – 96	0.92
4	97 – 128	0.15
5	129 – 160	0.86
6	161 – 192	0.50
7	193 – 224	1.14
8	225 – 248	1.68
AVG		0.99

5.2.1.2 *Experiment II: User Selection with GA*

In the previous experiment, ANNs are constructed with the random user data. In this section, thinking that utilizing data of people with similar features would give better learning results, the users are clustered by GA before the ANN learning. The people who responded to the questionnaire form the population. Therefore the size of the population (number of chromosomes) becomes 248.

The number of clusters is determined as eight, in order to have around 32 members in each cluster, if equally distributed, similar to the random user selection case. Within 200 iterations the GA has converged and formed the final clusters. Then, each cluster is trained with a separate ANN. The recommendation results for each of them are presented in Table 5.14.

Different than the ANNs with random users (Experiment I), now clusters have data from people with similar features. Therefore, the learning results are better than the learning performed with random user sets.

Table 5.14 Recommendation results for GA clustering (Experiment II)

Cluster	Number of People	Recommendation Error (%)
1	18	0.23
2	51	0.39
3	34	0.55
4	46	0.98
5	23	0.87
6	34	0.92
7	27	1.17
8	15	1.37
WEIGHTED AVG		0.77

5.2.1.3 Experiment III: User Selection with K-means Algorithm

K-means is one of the simplest and widely used unsupervised learning algorithms. Therefore, it is chosen as the comparison algorithm with GA.

In the equations in Section 2.5, the number of clusters (M) is eight and the length of the chromosomes (L) is 17. The results with the K-means clustering are shown in Table 5.15.

Comparing Table 5.14 and Table 5.15, one can say that clustering with GA produces better learning results (0.77 %) than clustering with K-means (1.27 %).

Table 5.15 Recommendation results for K-means clustering (Experiment III)

Cluster	Number of People	Recommendation Error (%)
1	40	1.00
2	37	1.63
3	29	1.03
4	27	0.68
5	23	2.39
6	28	1.04
7	41	1.21
8	23	1.40
WEIGHTED AVG		1.27

5.2.2 Automatic Clustering Experiments

In the previous sections, the number of clusters is chosen as eight. Here, it is demonstrated to find the number and the elements of the clusters automatically by three experiments (Experiment IV, Experiment V, and Experiment VI). In order to do this, the algorithms defined in Sections 2.4.2.2 and 4.2 are used.

5.2.2.1 Experiment IV: The Method of Bandyopadhyay and Maulik

This experiment is about to apply the automatic clustering method of Bandyopadhyay and Maulik (2002) given in Section 2.4.2.2 without any modifications. This means that unlike the proposed CCGA method, their algorithm does not have a penalizing transformation.

The experiment is repeated with several K_{max} values such as 10, 20, 50, 100, and 200. The expectation with this algorithm is to reach a value for the number of clusters which is less than K_{max}. However, in all experiments, the number of clusters converges to K_{max}.

This results in having a separate cluster and an ANN for each user which is not cost effective. Therefore, it is seen that this method does not provide a solution to the problem at hand.

5.2.2.2 Experiments V and VI: Automatic Clustering with CCGA

This section covers the experiments on CCGA clustering which is detailed in Section 4.2. The data set consists of 248 people who participated the questionnaire:

$$Data\ Set = \{X_1, X_2, \ldots, X_{248}\} \tag{5.1}$$

The minimum and the maximum number of centers are chosen as:

$$K_{\min} = 2 \tag{5.2}$$

$$K_{\max} = 100 \tag{5.3}$$

Then, the population size N is calculated as

$$N = 3\ (K_{\max} - 1) = 297 \tag{5.4}$$

The penalizing transformation limits n_{min} and n_{max} are taken as 10 and 30, respectively. The chromosomes include age, gender, and five genre preferences for weekday evenings, weekend daytime, and weekend evenings. According to this algorithm, the final number of clusters is resulted as 17, and hence, the population is divided into 17 clusters. Table 5.16 shows the distribution.

As expected, all cluster sizes are between the minimum and maximum limit values, n_{min} and n_{max}. This shows that CCGA has reached its aim with the penalizing transformation introduced.

Then 17, ANNs are formed and trained. Two approaches are taken for initialization of the ANN weights and results are compared. In the first one

(Experiment V), weights are initialized randomly between [-0.5, 0.5] interval and training starts with these random weight values. In the second approach (Experiment VI), the weight matrices are initialized using GA as detailed in Section 4.3. Table 5.17 gives the recommendation results for Experiment V after terminating ANN learning at 35^{th} epoch, and for Experiment VI, obtained after 23 ANN learning epochs.

Table 5.16 Clustering results with CCGA

Cluster	**Number of People**
1	15
2	14
3	22
4	17
5	11
6	10
7	15
8	17
9	10
10	11
11	15
12	15
13	16
14	13
15	11
16	17
17	17
TOTAL	248

The error values in Table 5.17 indicate that it is better to find the number of clusters automatically and form the clusters in an evolving way rather than using a pre-defined value for the cluster numbers.

When the results of Experiments V and VI are compared, it is seen that initializing ANN weights by GA gives similar error values in less epochs, i.e., less time than starting ANN learning with random weights. This is an important achievement for time critical operations like this one.

Table 5.17 Recommendation results for CCGA (Experiments V and VI)

Cluster	Recommendation Error (%) (Random Weight Initialization)	Recommendation Error (%) (GA-based Weight Initialization)
1	0.79	1.64
2	0.68	1.11
3	0.67	0.57
4	0.60	0.69
5	1.03	0.44
6	0.70	0.69
7	0.82	1.01
8	1.21	0.39
9	0.28	0.46
10	0.31	0.92
11	1.24	0.58
12	0.44	0.42
13	0.27	0.44
14	0.61	0.54
15	0.84	0.97
16	0.79	0.63
17	1.17	0.95
WEIGHTED AVG	0.74	0.72

5.2.3 Summary of Experiments with Questionnaire Data

Table 5.18 summarizes all six experiments conducted with questionnaire data. As explained in the previous sections, experiments differ according to the following attributes:

- whether the number of clusters is a pre-defined value or found automatically,
- whether clustering is performed randomly, by GA, K-means, or CCGA,
- whether ANN weights are initialized randomly or by GA.

It is clear from this table that the Experiment VI (data clustering by CCGA and ANN weight initialization by GA) is the best attempt to solve the problem at hand.

Table 5.18 Summary of experiments and results with questionnaire data

Experiments	Number of Clusters	Clustering	Weight Initialization	Average Error (%)
I	Pre-defined	Random	Random	0.99
II	Pre-defined	GA	Random	0.77
III	Pre-defined	K-means	Random	1.27
IV	Automatic	GA	N/A	N/A
V	Automatic	CCGA	Random	0.74
VI	Automatic	CCGA	GA	0.72

5.3 Recommendation Results with Recorded Data

In this thesis, eight people have provided real television watching data over a period of one month. Although each user provided 4-weeks of recorded data, since their starting day is not the same, there are total 36 different days of data.

Table 5.19 shows daily distribution of the recorded programs. According to it, no user has watched ten programs of the same genre in the first nine days. Finally, at the end of the tenth day, the learning starts since User5 has reached to at least ten "TD" and there are enough data for training and testing. Since the number of users is quite small, a single ANN would be capable of learning their data; therefore, the clustering step is not performed in these experiments.

In the incremental model explained in Section 4.4.3, the total number of users was not known at the beginning; therefore the ANN starts with one user input and expands when new users come. As shown in Table 5.20, at the 11th and 13th days the number of user inputs increase since the previous binary representation is not enough to represent the new number of users.

Table 5.19 Daily user and genre distribution

Day	User1	User2	User3	User4	User5	User6	User7	User8
10					TD			
11				Music	TD	Social		
12		Social		Music, TD	TD	Social		
13-15		Social		Music, TD	TD	Social	TD, Show	
16		Social		Music, TD	TD, News	Social	TD, Show	
17		Social		Music, TD	TD, News	Social, TD	TD, Show	Sports
18	TM	Social		Music, TD	TD, News	Social, TD	TD, Show	Sports
19	TM	Social, Sports		Music, TD, Social	TD, News	Social, TD, News	TD, Show	Sports
20-21	TM	Social, Sports		Music, TD, Social	TD, News	Social, TD, News	TD, Show	Sports, FD
22-25	TM	Social, Sports	News	Music, TD, Social	TD, News	Social, TD, News	TD, Show	Sports, FD
26-28	TM	Social, Sports	News	Music, TD, Social, News	TD, News	Social, TD, News	TD, Show, News	Sports, FD
29	TM, TD	Social, Sports	News	Music, TD, Social, News	TD, News	Social, TD, News	TD, Show, News	Sports, FD
30-32	TM, TD	Social, Sports	News, FM	Music, TD, Social, News	TD, News	Social, TD, News	TD, Show, News	Sports, FD
33	TM, TD	Social, Sports	News, FM, Education	Music, TD, Social, News	TD, News	Social, TD, News	TD, Show, News	Sports, FD
34-36	TM, TD, News	Social, Sports, News	News, FM, Education	Music, TD, Social, News	TD, News	Social, TD, News	TD, Show, News	Sports, FD

It should be noted that the weights causing the minimum test error during the current day's learning are stored to be used as the starting weights of the next day's learning, thinking that this method should give better results than randomizing the initial weights. Of course, for the days when the ANN expands, there will be additional user inputs whose weights to be introduced for the first time. Only these new weights are randomly initialized and the rest is carried over from the previous learning.

Then, the model is used to make recommendations. Out of 13 genre outputs, the one with the greatest value is considered as the recommended genre. Table 5.21 shows the learning results for the incremental approach, for both carry-over weights and random-start weights. The last day (Day 36) with random-start weights results in two recommendation errors which is 3.45 %. The ANN with carry-over weights for the same day, on the other hand, gives the perfect result with zero recommendation errors.

Figure 5.15 shows the learning curves for training and testing of the last day's ANN with carry-over weights. Around 100th epoch, the network hits the minimum test error.

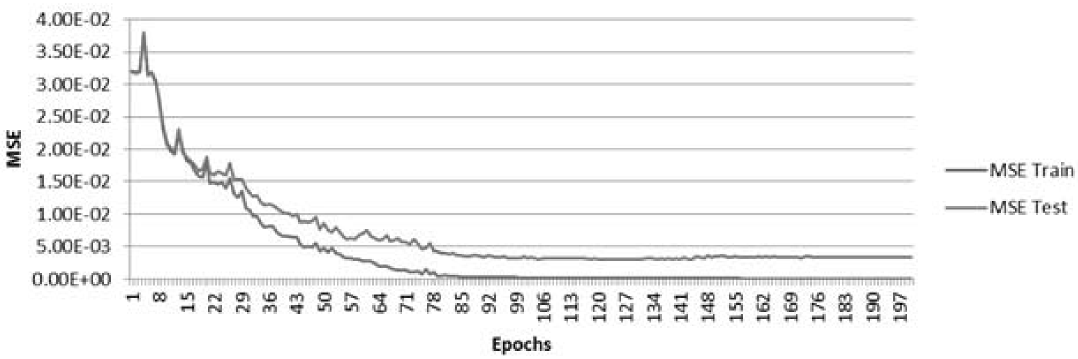

Figure 5.15 Train and test MSE for the ANN for Day-36.

It is seen also in Figure 5.16 that carry-over weights approach generally gives better results than random-start weights. Not only better results, but also it converges faster as expected (Figure 5.17).

Table 5.20 Number of user inputs vs days

Day	Number of Users	Number of User Inputs
1 – 9	0	0
10	1	1
11	3	2
12	4	2
13 – 16	5	3
17	6	3
18	7	3
19 – 36	8	3

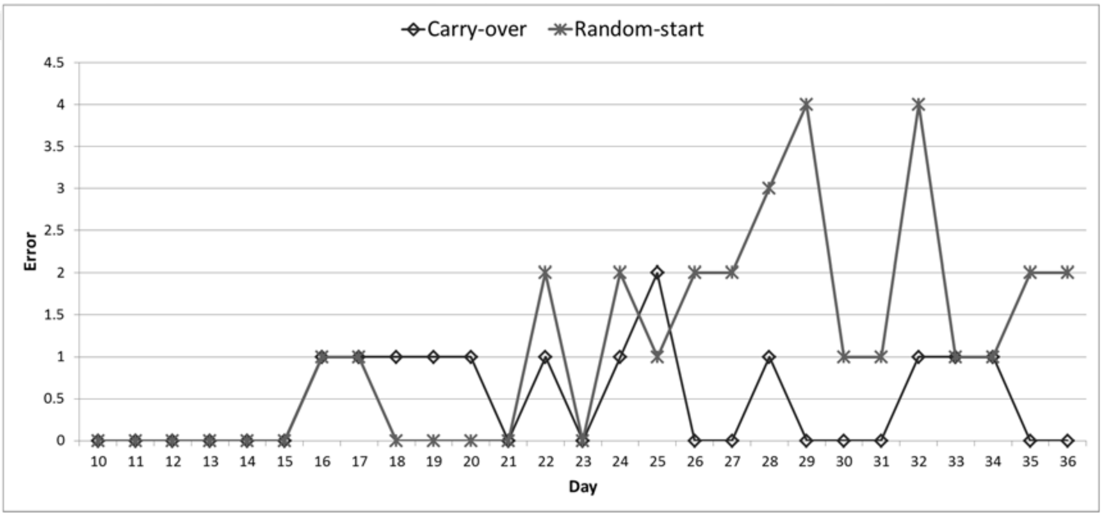

Figure 5.16 Recommendation errors of carry-over vs random-start weights.

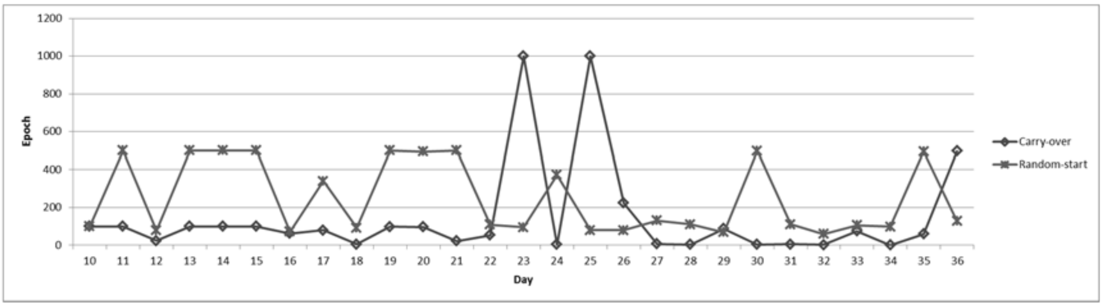

Figure 5.17 Epochs at ANNs converged; carry-over vs random-start weights.

Table 5.21 Incremental learning results

Day	Total Data	Train Data	Test Data	Recommendation Error (Carry-over Weights)	Recommendation Error (Random-start Weights)
10	11	10	1	0 (0.00 %)	0 (0.00 %)
11	35	32	3	0 (0.00 %)	0 (0.00 %)
12	60	53	7	0 (0.00 %)	0 (0.00 %)
13	88	79	9	0 (0.00 %)	0 (0.00 %)
14	96	87	9	0 (0.00 %)	0 (0.00 %)
15	105	95	10	0 (0.00 %)	0 (0.00 %)
16	122	110	12	1 (8.33 %)	1 (8.33 %)
17	152	138	14	1 (7.14 %)	1 (7.14 %)
18	173	155	18	1 (5.56 %)	0 (0.00 %)
19	226	203	23	1 (4.35 %)	0 (0.00 %)
20	252	227	25	1 (4.00 %)	0 (0.00 %)
21	270	244	26	0 (0.00 %)	0 (0.00 %)
22	291	261	30	1 (3.33 %)	2 (6.67 %)
23	305	275	30	0 (0.00 %)	0 (0.00 %)
24	327	296	31	1 (3.23 %)	2 (6.45 %)
25	354	319	35	2 (5.71 %)	1 (2.86 %)
26	380	342	38	0 (0.00 %)	2 (5.26 %)
27	402	363	39	0 (0.00 %)	2 (5.13 %)
28	421	380	41	1 (2.44 %)	3 (7.32 %)
29	446	401	45	0 (0.00 %)	4 (8.89 %)
30	471	423	48	0 (0.00 %)	1 (2.08 %)
31	489	439	50	0 (0.00 %)	1 (2.00 %)
32	508	456	52	1 (1.92 %)	4 (7.69 %)
33	531	478	53	1 (1.89 %)	1 (1.89 %)
34	551	495	56	1 (1.79 %)	1 (1.79 %)
35	562	505	57	0 (0.00 %)	2 (3.51 %)
36	571	513	58	0 (0.00 %)	2 (3.45 %)
				Total: 13	Total: 30

CHAPTER SIX
CONCLUSION

In this thesis, a novel system to collect program genres that users have watched and to make recommendations in accordance with their viewing habits is proposed by utilizing HbbTV. A complete method, including data collection at client side, intelligent learning at server side and program recommendation again at client side is proposed, implemented, and validated.

This method is proven by developing an HbbTV application. In practice, this application needs to be signaled on air by a broadcaster. Since this is beyond the focus of this work, the data from multiple users are also collected alternatively by a questionnaire and by manually recording watched programs on paper.

Here, the link between the collected data and program recommendation is filled with GA and ANN. However, the learning process can also be achieved by any other machine learning algorithm.

Due to its nature, the proposed method should work well with the increasing number of users. Therefore, a single ANN might not cope with the data coming from many users. To address this issue, the ANN learning is improved by clustering the users. The data of the users with similar features are clustered into smaller subsets and a separate ANN is created for each cluster. The number and the members of clusters are found by CCGA, a method based on GA proposed in this work, which improves an existing algorithm for cost and time critical processes.

Age, gender, and five first program genre preferences for weekday evenings, weekend daytime, and weekend evenings are used for clustering. Binary represented user ID, the hour of the day, the day of the week, time slot, gender, age, and the available genres at the time of concern are chosen as the inputs to the ANN. Genre information is the output vector, the one with the greatest value is chosen as the recommendation.

This proposed CCGA-based clustering system is tested with the data of 248 people. The optimum number of clusters is found as 17 and these clusters are formed by the CCGA. Then, 17 ANNs are trained and tested with the respective cluster data. The initial weights of these ANNs are calculated also by GA. The recommendation results with CCGA show that finding the number and the elements of clusters automatically gives smaller errors than using the pre-defined number of clusters.

With the recorded data, the incremental learning is found as the best approach, i.e., gathering the data from multiple users in parallel, starting learning whenever a user watches at least ten programs of the same genre so that training and testing sets can be formed as 90 % and 10 % of the data respectively, adding data from other users when they satisfy the same criteria and repeating this at the end of each day. This method yields 0 % error with data from 8 users over 36 days and converges quickly since the weights are preserved to be used as the starting weight values for the next day's learning.

The experiments prove that the best combination for this study consists of the following steps:

- Gather user information via an HbbTV application from digital receivers,
- Find the number and the elements of clusters automatically by CCGA,
- Construct a separate ANN for each cluster, initialize weights by GA,
- Train ANNs,
- Present program recommendations via the same HbbTV application,
- Repeat ANN learning periodically (e.g., every night),
- Repeat CCGA periodically (e.g., every week).

In this study, the main challenges of user profiling are addressed. The problem of data collection is solved by using an HbbTV application; the problem of initial profile generation for a new user is overcome by utilizing explicitly collected data to present recommendations; and, the problem of continuous adaptation of the profile for user's changing interests is solved by using artificial intelligence.

Since the broadcasters already have servers to host their current HbbTV applications to their thousands of users, it is assumed that it is feasible for the broadcasters to have powerful enough servers to handle the received data sent by the user devices as proposed in this work.

In order to avoid ANNs growing beyond control, the number of users for each network can be limited to a predefined value. When the relevant network reaches to the limit, clustering and learning are repeated, in addition to their regular times. Since the training and testing take place on the server side, parallel processing should not be a problem.

The method proposed here is tested by having a single user per receiver. However, in real life it is possible to have different viewers using the same receiver at different times of the day. To cover those cases, the HbbTV application can be written in such a way that it asks who is using the receiver at every time it is turned on with the option to register new users. Then, the server side can hold separate records for each user even if they share the same receiver.

As an addition to the method proposed here, more information can be given to the user via the HbbTV application. For example, along with the program recommendation based on user's viewing habits, the program that is being watched by most people could also be recommended.

Another improvement can be on the granularity of the genres used. In this study, 11 genres are taken from the European Standard (ETSI, 2016a). In addition, "Movie" and "Drama" genres are extended as "Foreign" and "Turkish", making 13 genres in total. It is also possible to get more detailed genre information (like "Football", along with "Sports") from the digital broadcast. Using this additional information, more relevant recommendations can be presented to the users.

In this thesis, the HbbTV technology is used to convey information on watched programs. Nevertheless, it can be used for some other purposes as well. Via HbbTV, broadcasters can have huge usage data at their server side, such as the most watched content via HbbTV catch-up application, the time of the day where HbbTV catch-up application is used the most, the most used HbbTV application, etc. These data can be modelled for better user experiences.

So far, a work similar to this thesis has not been seen in either industry or in scientific studies. Therefore, it is believed to bring a value in both areas. When compared to some of the well-known recent user profiling works, it is seen that the method proposed in this thesis has more advantages, such as not needing any proprietary way to collect data, any proprietary receiver software to run the algorithm, or any dependency to a PC (Table 6.1).

Table 6.1 Feature-wise comparison of this thesis with some other works

	Data Collection	**Proprietary Receiver Software**	**PC / Web**	**GUI**	**Heavy Process Location**
Barragans-Martinez et al., 2010	Web based	N/A	Yes	PC browser	Cloud server
Ardissono, Gena, et al., 2004	Proprietary	Yes	No	TV Screen	STB
Krstic and Bjelica, 2012	Proprietary	Yes	No	TV Screen	STB
Lee et al., 2010	Proprietary	Yes	No	TV Screen	Cloud server
Zhang et al., 2005	MHP (with proprietary additions)	Yes	No	TV Screen	Cloud server
This work	HbbTV	No	No	TV Screen	Cloud server

If this approach is applied by digital broadcasters and operators, all consumers having an HbbTV enabled digital receiver can benefit from getting smart recommendations regardless of the brands of their sets. Making smart recommendations to the TV users in order to provide a better TV viewing experience is the main motivation of this thesis. By using the method proposed here, digital broadcasters and operators can offer smart program recommendations to the owners of any HbbTV capable receivers without any software and hardware modification and they can handle many users simultaneously in a better way.

REFERENCES

Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, *17* (6), 734-749.

Alba, E. (2013). *Introduction on evolutionary algorithms.* Retrieved November 22, 2014, from http://neo.lcc.uma.es/opticomm/introea.html.

Alvarez, F., Martin, C. A., Alliez, D., Roc, P. T., Steckel, P., Menendez, et al. (2009). Audience measurement modeling for convergent broadcasting and IPTV networks. *IEEE Transactions on Broadcasting*, *55* (2), 502-515.

Ardissono, L., Gena, C., Torasso, P., Bellifemine, F., Difino, A., & Negro, B. (2004). User modeling and recommendation techniques for personalized electronic program guides. In L. Ardissono, A. Kobsa, & M. T. Maybury (Eds.), *Personalized digital television*, (3-26). Dordrecht: Springer.

Ardissono, L., Kobsa, A., & Maybury, M. (Eds.). (2004). *Personalized digital television*. Dordrecht: Springer.

Balabanović, M., & Shoham, Y. (1997). Fab: Content-based, collaborative recommendation. *Communications of the ACM*, *40*(3), 66-72.

Bandyopadhyay, S., & Maulik, U. (2001). Nonparametric genetic clustering: Comparison of validity indices. *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, *31* (1), 120-125.

Bandyopadhyay, S., & Maulik, U. (2002). Genetic clustering for automatic evolution of clusters and application to image classification. *Pattern Recognition*, *35* (6), 1197-1208.

Barragáns-Martínez, A. B., Costa-Montenegro, E., Burguillo, J. C., Rey-López, M., Mikic-Fonte, F. A., & Peleteiro, A. (2010). A hybrid content-based and item-based collaborative filtering approach to recommend TV programs enhanced with singular value decomposition. *Information Sciences*, *180* (22), 4290-4311.

Belew, R. K., Mcinerney, J., & Schraudolph, N. N. (1990). *Evolving networks: Using the genetic algorithm with connectionist learning*. Retrieved December 16, 2016, from http://nic.schraudolph.org/pubs/BelMcISch92.pdf.

Billsus, D., & Pazzani, M. (1998). Learning collaborative information filters. In *Proceedings of the 15th International Conference on Machine Learning*, 1998, 46-54.

Bjelica, M. (2010). Towards TV recommender system: Experiments with user modeling. *IEEE Transactions on Consumer Electronics*, *56* (3), 1763-1769.

Bjelica, M. (2011). Unobtrusive relevance feedback for personalized TV program guides. *IEEE Transactions on Consumer Electronics*, *57* (2), 658-663.

Brill, F. Z., Brown, D. E., & Martin, W. N. (1992). Fast genetic selection of features for neural network classifiers. *IEEE Transactions on Neural Networks*, *3* (2), 324-328.

Chang, E. I., & Lippmann, R. P. (1990). Using genetic algorithms to improve pattern classification performance. In *Proceedings of the 1990 Conference on Advances in Neural Information Processing Systems, 1990*, *3*, 797-803.

Cichocki, A., & Unbehauen, R. (1993). *Neural networks for optimization and signal processing*, (1st ed.). New York: John Wiley & Sons, Inc.

Cotter, P., & Smyth, B. (2000). PTV: Intelligent personalised TV guides. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and*

Twelfth Conference on Innovative Applications of Artificial Intelligence, 2000, 957-964.

Das, D., & Horst, H. (1998). Recommender Systems for TV. *Program*, (56), 4-5.

Das erste auf ihrem smart TV, (n.d.). Retrieved November 2, 2016, from http://www.daserste.de/specials/service/hbbtv-100.html.

Eberhart, R. C., & Dobbins, R. W. (1991). Designing neural network explanation facilities using genetic algorithms. In *Proceedings of 1991 IEEE International Joint Conference on Neural Networks*, 1991, 1758-1763.

ETSI. (2013). *ETSI TS 102 809 v1.2.1 Digital Video Broadcasting (DVB); Signalling and carriage of interactive applications and services in hybrid broadcast/broadband environments.* Sophia Antipolis, France.

ETSI. (2015). *ETSI TS 102 796 v1.3.1: Hybrid broadcast broadband TV*. Sophia Antipolis, France.

ETSI. (2016a). *EN 300 468 v1.15.1 - Digital Video Broadcasting (DVB); Specification for service information (SI) in DVB systems.* Sophia Antipolis, France.

ETSI. (2016b). *ETSI TS 102 796 v1.4.1: Hybrid broadcast broadband TV*. Sophia Antipolis, France.

Fondevila Gascón, J. F. (2012). La televisión conectada: ventajas e inconvenientes del estándar HbbTV. *Cuadernos de informacion*, *31*, 11-20.

Ghiglieri, M., & Tews, E. (2014). A privacy protection system for HbbTV in smart TVs. *2014 IEEE 11th Consumer Communications and Networking Conference, 2014*, 357-362.

Gutta, S., Kurapati, K., Lee, K. P., Martino, J., Milanski, J., Schaffer, J. D., et al. (2000). TV content recommender system. *Proceedings of the 17th National Conference on Artificial Intelligence*, 1121-1122.

Harp, S. A., Samad, T., & Guha, A. (1990). Designing application-specific neural networks using the genetic algorithm. In D. S. Touretzky (Ed.), *Advances in Neural Information Processing Systems 2*. (447-454). San Francisco: Morgan Kaufmann Publishers Inc.

Hu, C. H., Hsu, Y. Y., Hong, C. J., Hsu, S. H., Lin, Y. C., Hsu, C. M., et al. (2010). Home network management for IPTV service operations - A service provider perspective. *2010 IEEE/IFIP Network Operations and Management Symposium Workshops, 2010*, 1-7.

Illgner-Fehns, K. (2015). *Vision for HbbTV*. Retrieved August 21, 2016, from http://www.hbbtv.org/wp-content/uploads/2015/12/21_KlausIllgner_HbbTV_Presentation.pdf.

Isobe, T., Fujiwara, M., Kaneta, H., Morita, T., & Uratani, N. (2005). Development of a TV reception navigation system personalized with viewing habits. *IEEE Transactions on Consumer Electronics, 51* (2), 665-674.

Jabbar, H., Jeong, T., Hwang, J., & Park, G. (2008). Viewer identification and authentication in IPTV using RFID technique. *IEEE Transactions on Consumer Electronics, 54* (1), 105-109.

Jea, Y. T. (2000). *Basic concepts of data mining, clustering and genetic algorithms*. Retrieved March 10, 2016, from http://www.cse.buffalo.edu/DBGROUP/nachi/ecopres/tjea.ppt.

Kim, E., Pyo, S., Park, E., & Kim, M. (2011). An automatic recommendation scheme of TV program contents for (IP)TV personalization. *IEEE Transactions on Broadcasting, 57* (3), 674-684.

Krishna, K., & Murty, M. N. (1999). Genetic K-means algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, *29* (3), 433-439.

Krovi, R. (1992). Genetic algorithms for clustering: A preliminary investigation. In *Proceedings of the Twenty-Fifth Hawaii International Conference on System Sciences*, 1992, 540-544.

Krstic, M., & Bjelica, M. (2012). Context-aware personalized program guide based on neural network. *IEEE Transactions on Consumer Electronics, 58* (4), 1301-1306.

Kudová, P. (2007). Clustering genetic algorithm. In *Proceedings of International Workshop on Database and Expert Systems Applications, 2007*, 138-142.

Kumluca, A. (1997). *Neural network modeling of an ionospheric process: Temporal and spatial forecasting of the critical frequencies*. MSc Thesis, Middle East Technical University, Ankara.

Kurapati, K., Gutta, S., Schaffer, D., Martino, J., & Zimmerman, J. (2001). A multi-agent TV recommender. In *Proceedings of Eighth International Conference on User Modeling: Workshop on Personalization in Future TV*, 2001, 69-77.

Lee, S., Lee, D., & Lee, S. (2010). Personalized DTV program recommendation system under a cloud computing environment. *IEEE Transactions on Consumer Electronics, 56* (2), 1034-1042.

Lin, H. J., Yang, F. W., & Kao, Y. T. (2005). An efficient GA-based clustering technique. *Tamkang Journal of Science and Engineering*, *8* (2), 113-122.

Lu, Y., Lu, S., Fotouhi, F., Deng, Y., & Brown, S. J. (2004a). FGKA: A fast genetic K-means clustering algorithm. In *Proceedings of the 2004 ACM Symposium on Applied Computing*, 2004, 622-623.

Lu, Y., Lu, S., Fotouhi, F., Deng, Y., & Brown, S. J. (2004b). Incremental genetic K-means algorithm and its application in gene expression data analysis. *BMC Bioinformatics*, *5* (172), 1-10.

Maulik, U., & Bandyopadhyay, S. (2000). Genetic algorithm-based clustering technique. *Pattern Recognition*, *33*, 1455-1465.

Montana, D. J., & Davis, L. (1989). Training feedforward neural networks using genetic algorithms. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence* - *Volume 1*, 1989, 762-767.

Oren, Y., & Keromytis, A. D. (2014). From the aether to the ethernet - Attacking the internet using broadcast digital television. In *Proceedings of 23rd USENIX Security Symposium (USENIX Security 14)*, 2014, 353-368.

Schaffer, J. D., Caruana, R. A., & Eshelman, L. J. (1990). Using genetic search to exploit the emergent behavior of neural networks. *Physica D: Nonlinear Phenomena*, *42* (1-3), 244-248.

Sheikh, R. H., Raghuwanshi, M. M., & Jaiswal, A. N. (2008). Genetic algorithm based clustering: A survey. *2008 First International Conference on Emerging Trends in Engineering and Technology*, *2* (6), 2008, 314-319.

Sivanandam, S. N., & Deepa, S. N. (2008). *Introduction to genetic algorithms*. Berlin: Springer Berlin.

Smyth, B., & Cotter, P. (2004). Case-studies on the evolution of the personalized electronic program guide. In L. Ardissono, A. Kobsa, & M. T. Maybury (Eds.), Personalized digital television, (53-71). Dordrecht: Springer.

Topalli, I., & Kilinc, S. (2016a). Modelling user habits and providing recommendations based on the hybrid broadcast broadband television using neural networks. *IEEE Transactions on Consumer Electronics*, *62* (2), 182-190.

Topalli, I., & Kilinc, S. (2016b). Yapay sinir ağları ile kullanıcı profilleme ve kişiye özel TV program önerileri. In U. Guvenc & S. Birogul (Eds.), *Akıllı Sistemlerde Yenilikler ve Uygulamaları Sempozyumu 2016 Bildiriler Kitabı*, 193-197.

Whitley, D. (1995). Genetic algorithms and neural networks. In J. Periaux & G. Winter (Eds.), *Genetic Algorithms in Engineering and Computer Science*, (1st ed.) (191-201). New York: John Wiley & Sons, Inc.

Whitley, D., Starkweather, T., & Bogart, C. (1990). Genetic algorithms and neural networks: Optimizing connections and connectivity. *Parallel Computing*, *14* (3), 347-361.

Yang, J., Park, H., Lee, G. M., & Choi, J. K. (2015). A web-based IPTV content syndication system for personalized content guide. *Journal of Communications and Networks*, *17* (1), 67-74.

Zhang, H., Zheng, S., & Yuan, J. (2005). A personalized TV guide system compliant with MHP. *IEEE Transactions on Consumer Electronics*, *51* (2), 731-737.