electronics

MDPI

Article

An Improved Recommender System Solution to Mitigate the Over-Specialization Problem Using Genetic Algorithms

Oumaima Stitini *, Soulaimane Kaloun and Omar Bencharef

Computer and System Engineering Laboratory, Faculty of Science and Technology, Cadi Ayyad University, Marrakech 40000, Morocco; so.kaloun@uca.ma (S.K.); o.bencharef@uca.ma (O.B.)

* Correspondence: oumaima.stitini@ced.uca.ma; Tel.: +212-64-282238

Abstract: Nowadays, recommendation systems offer a method of facilitating the user's desire. It is useful for recommending items from a variety of areas such as in the e-commerce, medical, education, tourism, and industry domains. The e-commerce area represents the most active research we found, which assists users in locating the things they want. A recommender system can also provide users with helpful knowledge about things that could be of interest. Sometimes, the user gets bored with recommendations which are similar to their profiles, which leads to the over-specialization problem. Over-specialization is caused by limited content data, under which content-based recommendation algorithms suggest goods directly related to the customer profile rather than new things. In this study, we are particularly interested in recommending surprising, new, and unexpected items that may likely be enjoyed by users and will mitigate this limited content. In order to recommend novel and serendipitous items along with familiar items, we need to introduce additional hacks and note of randomness, which can be achieved using genetic algorithms that brings diversity to recommendations being made. This paper describes a **R**evolutionary **R**ecommender **S**ystem using a **G**enetic **A**lgorithm called RRS_{GA} which improves the fitness functions for recommending optimal results. The proposed approach employs a genetic algorithm to address the over-specialization issue of content-based filtering. The proposed method aims to incorporate genetic algorithms that bring variety to recommendations and efficiently adjust and suggest unpredictable and innovative things to the user. Experiments objectively demonstrate that our technology can recommend additional products that every consumer is likely to appreciate. The results of RRS_{GA} have been compared against recommendation results from the content-based filtering approach. The results indicate the effectiveness of RRS_{GA} and its capacity to make more accurate predictions than alternative approaches.

Keywords: genetic algorithms; recommender system; over-specialization; content-based filtering; limited content

Citation: Stitini, O.; Kaloun, S.; Bencharef, O. An Improved Recommender System Solution to Mitigate the Over-Specialization Problem Using Genetic Algorithms. *Electronics* **2022**, *11*, 242. https://doi.org/10.3390/electronics11020242

Academic Editors: Mehdi Elahi, Amin Beheshti and Mohammad Sina Kiarostami

Received: 8 November 2021
Accepted: 1 January 2022
Published: 13 January 2022

1. Introduction

Internet technology, particularly the World Wide Web, has advanced at an astounding rate. With this upgrade, there are new resources on the internet, such as documents, news, or articles to read, movies to stream, or items to purchase. Nowadays, the exponential growth of e-commerce websites and the advancement of the Internet of Things have made it difficult for shoppers to select correctly from the vast amount of offerings sold by these websites. People implicitly benefit from the features of recommender systems [1].

In this day and age of the information overload, it is quite difficult for users to find content that they are truly interested in. Users base their arguments on which movies to watch on their content, whether expressed in the form of communications data (genre, cast, or story-line) [2] or the feeling experienced after watching the corresponding movie trailer [3]. The media content has a significant impact on consumers' emotional affinity with the movie [4].

Many of the most extensive commerce platforms, such as Amazon.com, have also been using recommender services to help their clients look for things they want to buy. Many of

the world's leading sites, such as Netflix, have long used recommender services to help their users decide which movies to stream based on their personal preferences [5]. These programs provide search results that are customized to the user's preferences. When people visit a website, they are usually searching for things that are of interest to them. These things of interest may include a variety of topics. A recommender framework can also provide users with relevant knowledge about the items they are interested in. The ability to react quickly to changes in user preferences is a valuable advantage for such programs.

A recommender system is a processing method that measures the likelihood of a particular item being chosen by a specific person. The recommendation approaches are divided into eight categories as mentioned on Figure 1 based on how the recommendation is generated [6]: a collaborative filtering system, a content-based filtering system, a hybrid filtering system, a demographic recommender system, a knowledge-based recommender system, a risk-aware recommender system, a social network recommender system, and a context-aware recommender system.

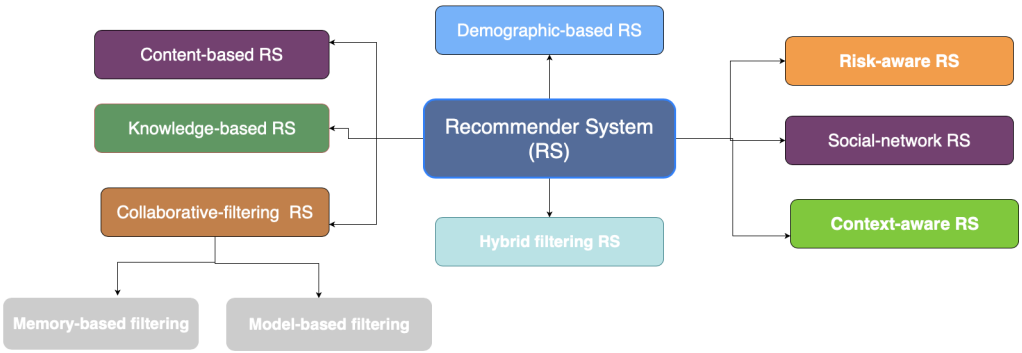

Figure 1. Different filtering techniques used in recommender systems.

A content-based recommender system creates a user profile by evaluating the characteristics of specific items to predict and produce recommendations [7]. As a result, the suggested objects are typically identical to items that the customer already enjoyed [8]. Collaborative filtering (CF) is a tool that focuses on user views being expressed. It is based on the "word of mouth" principle, which humans have historically used to shape an opinion about a good or service about which they are unfamiliar. The basic principle of this methodology is that the views of other users should be used to make a reasonable estimation of an active user's interest in a non-ranked object [3]. These methods presume that if users have similar preferences for a set of products, they are likely to have similar preferences for other things they have not yet rated. To produce recommendations, CF employs two distinct mechanisms: memory-based CF and model-based CF. The combination between these two precedent techniques generates the hybrid filtering approach [9]. A risk-aware recommendation system is one that is aimed at preventing and forecasting risks associated with an activity. The demographic-based technique implies that consumers with similar demographic profiles (e.g., age, gender, and country) would have similar interests. As a result, this recommendation system predicts various things based on demographic status. The knowledge-based recommendation system refers to a form of recommender system that is based on explicit knowledge about user preferences and recommendation criteria, such as which item should be recommended in which context. Social network recommender systems are characterized as a combination of social network data that can influence personal behavior and tag data [10]. Context-aware recommendations make use of contextual information in novel ways, such as user behavior, changing weather conditions, and cultural habits.

The content-based recommendation system lacks a critical tool for investigating anything unexpected. The system will recommend only objects with a high score as compared to the user profile. It is also known as the serendipity challenge, and it illustrates the limit of content-based advice, i.e., the over-specialization problem [11]. A "great" content-based

methodology would hardly provide anything novel, narrowing the variety of uses in which it would be helpful.

In that context, our focus in this research is on using a genetic algorithm to refine recommendations, add variety, and suggest new things that the user would enjoy. The contributions of this paper include developing a novel genetic algorithm for a content-based recommendation system that aims to select new items that the user will enjoy more. The genetic algorithm (GA) has been used to find the best suggestion list for a single individual based on their preferences. The proposed $\boldsymbol{RRS_{GA}}$ can solve the major challenge of content-based filtering, which is over-specialization and the limited content problems.

In this work, we attempt to fill several gaps. We present several contributions, including the following:

- We introduce the drawback of content-based recommender systems, especially the over-specialization problem.
- We provide an overview of genetic algorithms and their use in recommender systems.
- We propose a novel **R**evolutionary **R**ecommender **S**ystem based on a **G**enetic **A**lgorithm called $\boldsymbol{RRS_{GA}}$ that refers to using genetic algorithms to mitigate the limited content recommended to the user.
- We aim to demonstrate how genetic algorithms can brings diversity to recommendations being made.

The rest of this article is categorized in the following way: Section 2 contains our literature review about genetic algorithms and content-based filtering. We discuss our related works in Section 3. The proposed suggestion method, $\boldsymbol{RRS_{GA}}$, is seen and described in Section 4. Section 5 focuses on the testing of $\boldsymbol{RRS_{GA}}$ and the analysis of the findings. We mention the discussion of our work in Section 6 . Finally, Sections 7 and 8 bring the article to a close and discusses possible further research.

2. Literature Review

This study focuses on a genetic algorithm-based recommendation method [12] that addresses the issue of over-specialization in content-based algorithms. This literature review aims to provide a summary of the previous research in the subject fields. Recommender systems enable the task of recommending or exposing new and current products to a customer who has never seen them before. Recommender systems use a variety of algorithms and techniques to produce suggestions. The most common systems are collaborative filtering [13], which considers other users in the scheme [14], and content-based filtering, which focuses solely on the content. This literature review aims to provide an overview of previous research in these subject fields.

This literature review can broadly be divided into the following categories: Section 2.1 Content-Based Filtering, Section 2.2 Content-Based Filtering Limitations, and Section 2.3 Theoretical Background of Genetic Algorithms.

2.1. Content-Based Filtering

The system learns to make recommendations by analyzing the similarity of features between items [15]. For example, based on a user's rating of different movie genres, the system will learn to recommend the genre that is positively rated by the user [8]. A content-based recommendation system builds a user profile based on the user's previous ratings. A user profile represents the user's interests and can adapt to new interests [4,5].

Information retrieval, analysis, and filtering are the foundations of the content-based filtering approach [16]. This method is most often used in areas where content can be read or analyzed, such as news articles, movies, and everything else containing metadata. It also makes suggestions based on what the user has already seen. Labels may be used to classify the contents, and each label is assigned a weight based on how well it describes the article. Nearest-neighbor or clustering algorithms may be used to suggest other articles to the active user based on these labels and user expectations. This system, however, is challenged by new users with limited knowledge and a limited number of labels.

2.2. Content-Based Filtering Limitations

The content-based filtering strategy is most commonly utilized in situations where material can be read or examined, such as news articles, movies, and anything else containing metadata. It also makes suggestions depending on what the user has already watched. Labels can be used to describe the contents, and each label is assigned a weight based on how effectively it describes the item. Nearest-neighbor or clustering algorithms can be used to propose other articles to the active user based on these labels and user preferences. This strategy, however, is challenged by new users who have minimal information and a restricted number of labels. The content-based recommender system has solved the challenges of collaborative systems, but it has some drawbacks, including:

- Limited Content: Content-based techniques have a limit on the quantity and kind of characteristics shared with items that they propose manually or automatically. Domain expertise and taxonomies for the specific domain are also required. If the evaluated content lacks sufficient information, the content-based recommender system will be unable to make appropriate suggestions.
- Lack of Serendipity: In a content-based recommendation system, there is no crucial technique for discovering anything unexpected. Goods are advised based on their high score while matching the user profile; as a result, the user suggests items that are comparable to previously rated items. This is also known as the over-specialization problem, and it highlights the tendency of content-based systems to offer ideas with a limited degree of inventiveness. To find some innovative and surprising recommendations, a great content-based approach is required.
- Over-Specialization: A content-based recommendation system lacks a necessary approach for investigating anything unexpected. Only goods with a high score when compared to the user profile can be recommended by the system. It is also known as the serendipity issue, because it illustrates the limit of content-based suggestions. A "perfect" content-based method would offer little new content, limiting the variety of applications for which it would be useful.
- New User: A significant number of ratings must be collected in order to create a recommendation system that can learn about user preferences. Due to the fact that no previous data is available, the system is unable to make trustworthy suggestions to new users.

2.3. Theoretical Background og Genetic Algorithms

2.3.1. Principles: Definition and Vocabulary

A genetic algorithm attempts to answer the challenge of how a computer can solve a problem without being explicitly told how to solve it. Genetic algorithms attempt to bring ideas from evolution and natural genetics to computers. The goal of genetic algorithms is to construct computer programs that model and reproduce the evolutionary process of nature. Genetic algorithms solve an optimization issue by converting a population of alternative solutions. The algorithms operate on the properties of the issue rather than the problem itself. The values of the solutions are compared, and the greater the value, the more likely that solution is to advance to the next level.

Genetic algorithms do not employ variables but rather associate them with a specific coding; variables are considered using characters representing a sequence of codes. Each variable is converted into a gene, which can include one or more codes that can express various characteristics. An individual is represented by the code sequence. In other words, a chromosome is a possible solution. The goal of the treated problem is described using a function that permits assessing the likelihood that an individual will be picked or not in order to recreate new solutions. The fitness function is the name of this function. Limitations are considered in the fitness function by punishing people who disobey the problem's constraints. The exploration of the space of potential solutions is based on two techniques that aim to create new solutions at random from the beginning population. The genetic operators consist of crossing and mutation mechanisms. The selection process aims to

guide the investigation by identifying the individuals who have the best chance of getting selected.

In order to explain how genetic algorithms work, Figure 2 presents the different steps of a simple genetic algorithm.

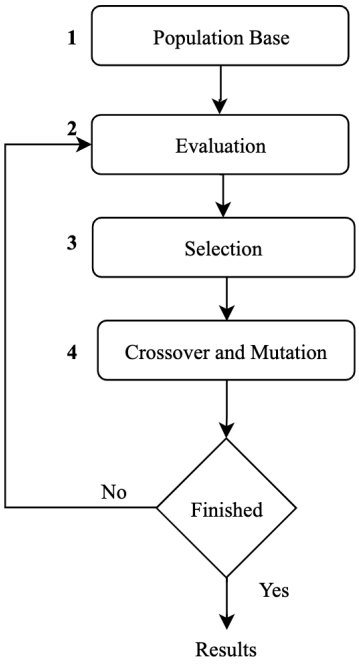

Figure 2. The main steps of a genetic algorithm.

2.3.2. Initial Population

Any genetic algorithm [17] is built around a solution represented by its chromosomes [18] which are then passed on to their offspring [19]. In GAs [20], it is common to represent each solution with a binary string [21]. Each bit indicates whether the solution has a particular characteristic or not and uses numbers to denote the strength of a function in a solution. Using numbers rather than single bits to show the existence of a feature has the advantage of allowing one to identify a solution as being better than others by using more than just the feature's presence.

$$[1, 2, 3, 4, 5] \quad (1)$$

In Equation (1) we present a possible solution by describing the strengths of each feature. The solution contains five genes, each one represents a specific movie.

2.3.3. Fitness Function

The fitness function is a function that assigns a score to a given individual. We found that an individual with a high chance of being closest to the perfect solution will receive a higher score than others [22]. In GA, the fitness function is crucial, because it determines whether a solution is successful or not, and a poorly constructed fitness function can result in less-than-ideal solutions.

In fitness algorithms, utility functions or simple mathematical functions, such as number and average, can be used. Utility functions measure the importance of something to an individual, rather than just plugging it into an algorithm.

2.3.4. Genetic Algorithm Operators

Crossover, mutation, and selection are the three operators that make up a genetic algorithm. Each operator has a distinct and equally essential function to play.

Crossover

Crossover is a technique for changing chromosome programming from one generation to the next by producing children or offspring. These offspring are generated using parent chromosomes (generated chromosomes). There are many methods for producing offspring, including a single-point crossover operator, a multi-point crossover operator, and a uniform crossover operator:

- Single-point crossover: A crossover point is created at random in a single-point crossover, determining how parents share information to form children. The crossover point is 2.
- Multi-point crossover: Multiple crossover points are randomly generated in a multi-point crossover, determining the points for knowledge sharing between parents to form children. As a consequence, information is exchanged between the crossover points.
- Uniform crossover operator: Knowledge is exchanged between parents in the uniform crossover depending on specific probability values. A probability matrix of the same length as the parents is created at random. If the probability value at one or more of the indexes approaches a predefined threshold, knowledge is shared between parents at such indexes to form children.

Mutation

In the genetic algorithm, the mutation operator is a widespread technique. Using hybridization, it has made its way into other heuristic and meta-heuristic strategies. Its potential to bypass local optima and seek a larger solution area is the main reason for this.

- Random Mutation: Random numbers are used to modify genes on a chromosome at one or more positions.
- Flip Mutation: The maximum values of all genes are used to modify genes from a chromosome at all positions. The maximum value for a binary chromosome is 1.
- Bit-String Mutation: This is a flip mutation variation in which random numbers calculate gene alteration for one or more roles.
- Boundary Mutation: If the value of a gene falls below the given lower bound or exceeds the given upper bound, the gene is updated.
- Swap Mutation: Two genes on the same chromosome switch positions, i.e., gene values are swapped.
- Inverse Mutation: This is a swap mutation variant. Genes that are equally spaced around the middle of a chromosome are exchanged, causing the chromosome to reverse.
- Insert Mutation: A chromosome gene is cropped from one location and inserted into another. Alternatively, more than one gene may be cropped at various locations and placed at a different location.
- Shift Mutation: By transferring genes to the left or right N times, one or more genes on a chromosome are changed.
- Increment or Decrement Mutation: The values of genes on a chromosome can be changed using ratios to increase or decrease their values.

Selection

Selection determines which individuals between the old and offspring populations will form the next population, which will be used for crossover and mutation to create the next generation of offspring. The selection operator represents the main pillar of genetic algorithms. There are many methods use to describe the selection step: elitist selection, random selection, and tournament selection.

- Elitist selection: This is a selection method that only selects individuals with the best (highest) fitness values. A small number of individuals with the highest fitness values is chosen to pass on to the next generation while avoiding the crossover and mutation operators. Elitism avoids the random elimination of individuals with good genetics

by crossover or mutation operators. The population should not have too many elite individuals, or else the population will tend to degenerate.
- Random selection: This method combines the populations and randomly chooses N item to obtain the new population. Tournament selection is also a popular literary technique since it may function with negative fitness values.
- Tournament selection: Tournament selection is a selection strategy used in a genetic algorithm to choose the fittest candidates from the current generation. These chosen candidates are then handed on to the following generation. K individuals are chosen and a competition is run among them in a K-way tournament selection. Only the fittest candidate among those selected is picked and handed down to the next generation.

3. Related Work

3.1. Content-Based Filtering

Reference [4] provides a deep evaluation on how content-based recommendation algorithms can be influenced by the metadata information used in the domain of movies and television program descriptions. They found that the greater the number of metadata attributes used in combination, the better the performance.

Reference [8] proposes a system that only uses the content data of movie as the training dataset. They suggest a method that extracts features from the content of movies and transforms the textual content data into feature vectors which can semantically keep linear relationships .

Reference [5] proposes a system which provides a movie recommendation based on the genres of the movie. If a user highly rates a movie of a particular genre, movies containing similar genres will be recommended to them.

3.2. Using Genetic Algorithms in the Recommender System

The GA has been used in three different ways in RS [23]: clustering [24], hybrid models [9], and using GA without the need for the extra details offered by the hybrid model [2,10,13,14,25].

The clustering approaches for RS that have been proposed so far have strengths and disadvantages. Their features may significantly affect the precision of the system's recommendations for the active user. Reference [24] presents a hybrid evolutionary-based method for item clustering and production of the most suitable clusters in the offline phase of the collaborative filtering Recommender System (RS). They propose a method that combines the Genetic Algorithm (GA) and the Gravitational Emulation Local Search (GELS) algorithm to measure the quality of data clustering with the proposed method and thus to increase the accuracy of the system-generated suggestions.

Reference [9] proposes a system which recommends relevant data to the user according to their preferences and history involving movies. They combine the content, collaborative techniques, and some demographic information into a hybrid approach, where additional content features are used to improve the accuracy of collaborative filtering with the genetic algorithm to provide recommendations to the user.

Reference [26] proposes a hybrid recommender system based on GAs and the collaborative filtering technique. They suggest a system that integrates data from various sources (product, customer, and transaction data) to form the customer preference profile and apply a GA to optimize a vector of the feature weights, which are used to measure the similarity among customers.

Reference [27] proposes a combination between the genetic algorithm and the k-nearest neighbor algorithm, k-NN . They present a crack classification model for a spherical tank based on an optimal hybrid feature pool creation. They suggest 18 features from different domains for the hybrid feature pool, and then they apply a heuristic search-based genetic algorithm to remove the data redundancy and to reduce the dimensions of the original hybrid feature pool. The genetic algorithm process used extracts two optimal feature

subsets which are provided to a k-NN classifier for data classification. They achieved 99.8% in the accuracy metric.

Table 1 summarizes the reviewed contributions regarding the use of genetic algorithm in recommendation systems.

Table 1. Summary of contributions.

Contribution	Method	GA Usage	Handled Issues
[13]	GA	Find the optimal similarity matrix.	Cold-start problem.
[14]	GA	Select the best list of items that meets active user interests.	Cold-start and sparsity.
[25]	GA	Improve the accuracy of multi-criteria recommender systems.	Multi-criteria recommendation.
[24]	GA, Clustering	Produce the most suitable clusters in the collaborative filtering RS.	Data clustering in CF systems.
[28]	GA, Clustering	Improve clustering method by changing the fitness function in the GA.	Cold-start and sparsity.
[29]	GA, Clustering	Generate the cluster-based optimal ranking.	Improve precision of search results.
[9]	GA, K-NN	Recommends relevant data to user according to their history preferences.	Sparsity.

Reference [14] suggests a novel genetic-based recommender system (BLIGA) that depends on semantic information and historical rating data. They present a multi-filtering level collaborative-filtering technique, BLIGA. The proposed approach, BLIGA, produces recommendations to active users using the genetic algorithm approach. The main idea is to search for a list containing highly correlated items, which items two characteristics: rated by the neighbors of AU and potential favorite items of high predicted values of ratings, in terms of semantics.

4. Proposed Approach

4.1. Over-Specialization Problem

All recommendation systems will inevitably face the over-specialization problem. The over-specialization issue is described as having restricted content and producing suggestions with a low level of novelty and a lack of a method for discovering something unexpected to recommend to the user. When a user likes a new object, the algorithm only recommends related things that the user has already enjoyed as illustrated in Figure 3.

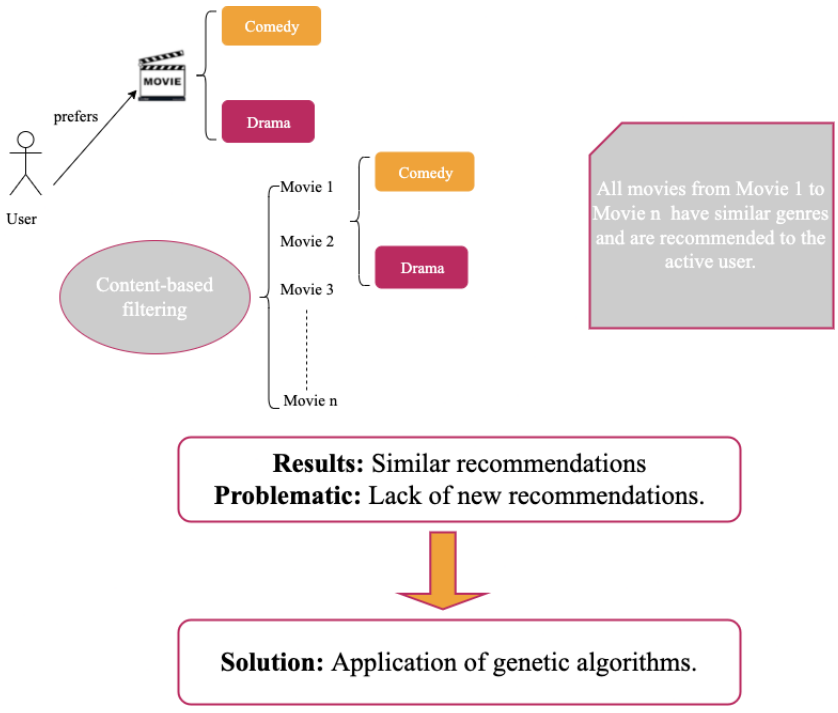

Figure 3. The over-specialization problem.

The content-based filtering algorithm has no built-in mechanism for discovering anything unpredictable, which gives rise to the over-specialization problem. Compared to the user's profile, the algorithm recommends objects with high ratings, implying that the user would be offered items that are close to those that have already been scored.

This limitation is also known as the serendipity issue, and it refers to the inability of content-based systems to generate suggestions with a low level of novelty. For example, if a user has only rated Stanley Kubrick films, only such films would be recommended. A "great" content-based approach will hardly discover something different, limiting the number of applications it could be used for.

Content-based systems suffer from the over-specialization problem, because they only suggest products close to those that users have already scored. The implementation of any randomness may be one solution to this problem [30]. Furthermore, over-specialization is not simply the problem that content-based programs cannot suggest things that are not similar to what the user has already seen. Things that are very close to what the user has already encountered, such as a separate news reports explaining the same incident, should not be suggested in certain situations.

Serendipity in a recommender system is the idea of getting an unlikely and serendipitous article recommendation. It is a way to diversify recommendations [16]. Although people depend on chance and experimentation to discover new articles they did not know they needed, content-based programs lack an essential method of delivering serendipitous suggestions due to over-specialization. In conclusion, the adoption of strategies for realizing operational serendipity is an effective way to extend the capabilities of content-based recommender systems to mitigate the over-specialization problem by providing the user with surprising suggestions.

On the one hand, limited content analysis indicates that the system can only provide a small amount of knowledge about its users or the content of its products. On the other hand, over-specialization results from the way content-based programs recommend new items. A user's expected ranking for an item is high if the item is close to the ones the user enjoys. For example, in a movie suggestion system, the system may suggest a movie to a user in the same genre or that stars the same actors as ones that the user has already seen. As a result, the system can miss out on unique and attractive objects to the user.

In that regard, our aim in this study is to use an optimization method, especially the genetic algorithm in content-based filtering, to achieve the aforementioned aim. A genetic algorithm aims to solve how a machine can solve a problem without being specifically told how to solve it. Genetic algorithms try to apply biology and natural genetics to programming. This encourages one to let genetic programming build the framework to solve the problem rather than specifying it.

Genetic algorithms work similarly to genetic programming, only that, instead of dealing with whole systems, GAs only work with constructs that need to be streamlined. A population of possible solutions is converted into an optimization problem by genetic algorithms.

Our strategy was to create a flexible recommendation system that can provide recommendations to users based on a their preferences. We developed a revolutionary algorithm based on the idea that recommender systems should help users find products that are outside of their immediate inclinations and novel items that they did not know of before. That is, we provide recommendations based on the interest of the user using genetic algorithms.

This will allow us to produce recommendations to the user with a high degree of novelty based on their preferences. Our algorithm will learn and evolve from the user preferences easily in the early stages to alleviate the over-specialization issue as demonstrated in Figure 4.

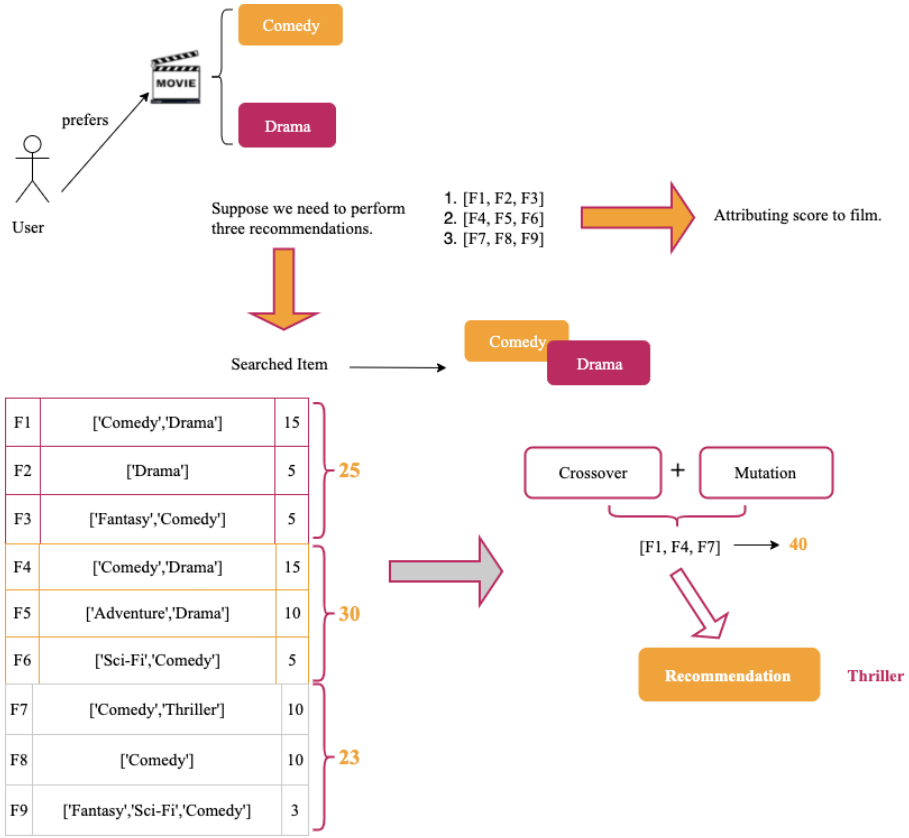

Figure 4. The genetic algorithm architecture.

4.2. The Proposed Genetic-Based Recommender System

Instead of choosing specific products to form a recommendation list, RRS_{GA} focuses on the overall content of the recommendation list. The main principle is to review the entire suggestion list hierarchically and show new products to the customers that may interest them. Algorithm 1 presents the main procedure of RRS_{GA}.

Algorithm 1 The main procedure of RRS_{GA}.

Input: List of film genre.
Output: Recommendation List.

```
1:  Generate the initial population (initPop) that contains M individuals. Each chromosome
    contains N random genotype where each of those genotype is a movie id.
2:  Get the genre of the movie.
3:  Get the genre score.
4:  for elem in population do:
5:      for item in elem do:
6:          Call getGenre function.
7:          Call GetScore function.
8:          Select the scores of the entire population.
9:  Apply crossover operator with uniform method.
10: Apply mutation operator with random method.
11: end
12: Select the best individual and recommended its items to the user.
```

1. **Initialize population (Line 1).** Initially, RRS_{GA} fills the population with M randomly generated lists, initPop. Each chromosome contains N random genotypes where each of those genotypes is a movie ID.
2. **Fitness function definition (Lines 2–10).**

- Step 1: Obtain genre of the movie
  Select the genre based on movie ID.
- Step 2: Obtain genre score
  The core idea of this phase is to attribute a score for each solution. For that, if the searched item has, for example, "Action" as the first genre and "Sci-Fi" as the second genre, a similar solution will gain 15 points as a score. Otherwise, the solution which matching the first genre gains ten points, and the process continues in that same way.
- Step 3: Fitness values
  In this step, we generate the scores by looping over a chromosome. At the end of this phase, we will obtain the scores of the entire population.

$$[13, 441, 227, 593, 99] \tag{2}$$

$$[15, 10, 10, 5, 2, 2] \tag{3}$$

For example, if individual in Equation (2) generates scores of Equation (3), the sum of the scores would be 44. If we ran through the same process on the other members of the population, the results would differ.

3. **Apply genetic algorithm operators (Lines 9–10).**
   - Crossover
     In the crossover step of the algorithm, the offspring are generated with features from both parents. This allows for the development of offspring with higher affinity qualities for those genres. For this step, we use the uniform crossover method. Each parent's section would have a matching segment of the same length and size as the other parent. The child would then receive elements from one of the parents at random. The chosen elements are joined together to make the offspring. The child will be the same length as both parents but will have different characteristics.
   - Mutation
     We added a little variety into the population via the mutation mechanism. A mutation is a process of altering the offspring generated by existing solutions to add new and valuable features while removing neither worthwhile nor helpful features. For this step, we use the random mutation method.
   - Selection
     We chose the elitist selection method as it defines which individuals from the old and offspring generations make up the next population.
4. **Select the optimal recommendation list (Line 12).** $\boldsymbol{RRS_{GA}}$ captures the items of the selected individual and recommends it to the user.

4.3. Methodology and Overall Approach

We started with a basic recommendation engine that lists all similar movies based on what the user wants. It performs the primary recommendation using a content-based filtering algorithm. For example, if the user already liked a specific film from the "Action" genre, the principal content-based recommendation recommends all films that have a similar genre to the user. Figure 5 presents the architecture of $\boldsymbol{RRS_{GA}}$.

For that, our proposed technique contains a four-phase process to deal with this problem.

- **Phase 1: Content-based algorithm**
  We created a similarity matrix using a cosine similarity function between movies, and then we recommended the most similar movies by searching on the matrix table for all movies with a score of more than 0.80 .
- **Phase 2: Genetic algorithm**

We needed to incorporate additional hacks and take care of randomness to suggest novel and serendipitous items alongside common items, which could be accomplished by leveraging genetic algorithms that carry a variety of recommendations.

1. Initial Population
   The movies in our MovieLens dataset are divided into 18 distinct genres. Since we only want five recommendations, the vector size should have five values, each of which corresponds to a different movie ID to reflect each individual. Firstly, we need a sample population for our genetic approach. The sample population in the GA is chosen from the whole population as N random solutions. The solution is chosen at random, since each chromosome comprises five movie IDs. As a result, the algorithm starts with actual data and finds the best solutions. Our initial population should be based on existing movies, since we will need to find neighbors between them. We will construct the initial population randomly. The size of the initial population will be 1947. We will have a population of 1947 individuals with preferences for the five different movies in the system. The reasoning behind choosing our population in this way is that our ultimate goal is to find the best individual who will find neighbors who have the same preferences requested by the user with a new element that will create serendipity. Individuals will be used to illustrate how our initial population will look. For this reason, the table only includes five individuals. The individual's representation will be of format mentioned in Equation (4) to indicate the different possible movies in the system.
   $$[1, 2, 3, 4, 5] \tag{4}$$
2. Fitness Function
   In our example (see Table 2), we assume the user liked a movie with "Adventure" as the first genre and "Animation" as the last one.
   $$[574, 4964, 31, 297, 60, 389, 1097] \tag{5}$$
   Regarding the first individual described in Equation (5): The first movie in this individual is 574, and it has the genres Adventure, Animation, and Children. They both preferred Adventure as the first genre and Animation as the second so that the score could be set at 10.
   Table 2 below shows a representative sample of people and the health ratings assigned to them.

Table 2. Fitness scores for individuals.

Individuals	Fitness Score
(574, 4964, 31,297, 60,389, 1097)	70
(5028, 6942, 103,755, 139,130, 30)	75

- **Phase 3: Genetic algorithm operators**
  - Crossover: The algorithm's crossover process is the process by which the offspring are created with characteristics from both parents. For that step, we will use the random crossover method. Each gene (bit) is chosen at random from one of the parent chromosomes' corresponding genes. To make the offspring, the chosen segments are joined together. For crossover, we use a uniform crossover method with an N/10 probability, as we want only N recommendations. For example if we assume that we need five recommendations, we would use a uniform crossover method with a 5/10 probability.
  - Mutation: We added a little variety into the population via the mutation mechanism. For mutation, we will use the random mutation method. For mutation, we could have a 5% chance that after crossover one of the genes is randomly

selected and changed to a random movie ID number. At the end of this step, we obtained the new offspring population, and we again calculated their fitness scores, and then we retained the best performing individuals between the old and new populations.

– Selection: We used the elitist selection method, as it defines which individuals from the old and offspring generations make up the next population.

- **Phase 4: Recommendation**
  This phase is the last one to provide the optimal recommendation list, it show the chromosome with the best fitness. This final step suggests an unexpected and fortuitous thing to the user. The active user has a good luck in finding new items that are likely to be of their interest.

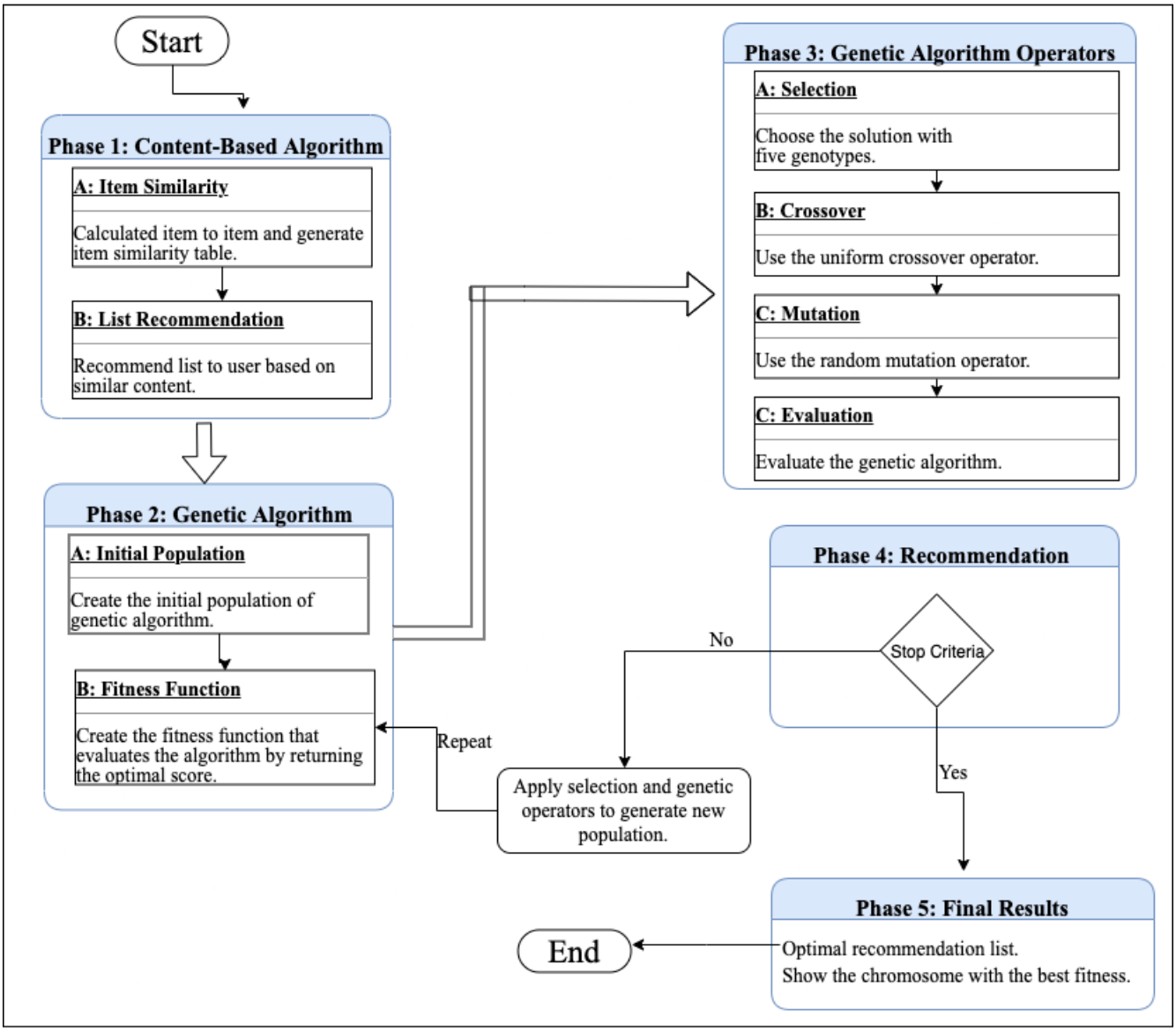

Figure 5. The main architecture of RRS_{GA}.

5. Evaluation and Results

Addressing a problem and using an evaluation method to see how the problem has been solved is part of evaluating feedback systems. Recommender systems must have a

solution to a problem in order to be helpful. The problem must be clearly described to determine whether it has been solved or not.

This section explains how the suggested RS, $\boldsymbol{RRS_{GA}}$, was put to the test. Several tests were carried out to compare the suggested RS to other suggestion techniques:

– Content-based filtering: This recommendation technique generated the recommendations based on the cosine similarity.
– The use of clustering: In reference [7], they use Clustering, especially the k-means algorithm to enhance the recommendation.

The section below contains information about the datasets that were used in the tests. Section 4.2 goes into the experimental approach that was used and the measurement criteria that were used. Finally, in Section 4.3, the findings are presented and discussed.

5.1. Experimental Data

The benchmark and synthetic datasets have been used in all experiments. MovieLens [31] is a dataset acquired by the GroupLens study group at the University of Minnesota for use in recommender method research. To secure the identities of the users, the data is anonymized. Movies, users, and movie ratings are all included in the information. The dataset in question was selected because it is widely used in the field for benchmarking recommender systems [32]. There are approximately 6040 users, 3900 movies, and 100,000,209 reviews in the dataset. Table 3 summarizes the statistics of the experimental data. The potential scores are 1, 2, 3, 4, and 5, with 5 suggesting that the user enjoyed the film the most, and 1 indicating that the user enjoyed the film the least.

Table 3. Specifications of the used MovieLens dataset.

Properties	MovieLens Dataset
Number of users	6040
Number of movies	3900
Number of genres	18
Number of reviews/ratings	100,000,209

5.2. Experimental Design

The $\boldsymbol{RRS_{GA}}$ evaluation process consists of four different experiments that were conducted using a MacBook Pro 16GB RAM (Apple, Cupertino, CA, USA). The first experiment aimed to evaluate first the content-based recommendation algorithm. The second experiment examined the data to evaluate the selected recommendation list of $\boldsymbol{RRS_{GA}}$. In both cited experiments, we considered the obtained results using the MovieLens dataset. The third experiment compared the results of the proposed approach $\boldsymbol{RRS_{GA}}$ with the results found by the content-based filtering algorithm. The CPU time needed by $\boldsymbol{RRS_{GA}}$ to predict the recommendation was compared with the required CPU time of the content-based technique. The last experiment assessed the $\boldsymbol{RRS_{GA}}$ in terms of recommendation quality using the criteria of *Recall*, *Precision*, *Diversity*, *Serendipity*, and *Novelty*.

In all content-based applications, the recommendation system does not predict the user's preferences of items, such as movie ratings, but tries to recommend items to users that they may prefer and use [33]. In this case we were not interested in whether the system properly predicts the ratings of these movies, but rather whether the system properly predicts that the user will add these movies to their queue (i.e., use the items).

When recommending items to users, various metrics must be considered, not only the accuracy of a recommendation prediction. There are several measures and metrics for evaluating recommendations.

- Recall: The recall is the percentage of the favorite recommended items to the total favorite items of the active user, where TP is true positive and FN is false negative (Equation (6)).

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

- Precision: The precision is the ratio fraction of the interesting recommended items to the total number of recommended items, where FP is false positive (Equation (7)). Both recall and accuracy are affected by the number of recommended items (N). As a result, the recommendation quality metrics were computed using a different number of suggested items.

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

- Diversity: Diversity refers to the inclusion of several types of items in a user's recommendation that vary from their previous preferences. The center list similarity metric is used to calculate diversity. The diversity metric is a measure of how distinct and various recommendations differ from one another. Consider a consumer who just finished watching the first movie of a trilogy on Netflix. A low diversity recommender would recommend only the following movies of the trilogy or films directed by the same director. High diversity, on the other hand, can be accomplished by suggesting items at random, which is why the use of genetic algorithms is beneficial in bringing diversification to the recommendations (Equation (8)).

$$Diversity = 1 - Similarity \tag{8}$$

- Serendipity: Serendipity is a metric that measures how unexpected or relevant recommendations are provided to the user. This equation is divided into two parts: the degree of surprise and the user's relevancy. Surprise is calculated as the difference between the likelihood that an item i is recommended for a user and the likelihood that an item i is suggested for any user. The chance of a recommendation is just a function of its overall rank among n items (Equation (9)):

$$P_i = \frac{n - rank_i}{n - 1} \tag{9}$$

A recommendation is considered meaningful to the user if the user evaluated it highly. When we put all of the information together for a user, we generate Equation (10):

$$Serendipity_u = \frac{1}{n} \sum_{i \in n} max(P_i(user) - P_i(allUser), 0) \times rel_i(user) \tag{10}$$

Global serendipity is just the average serendipity across all users. The most challenging part of this equation is determining relevance. If a user does not rate a significant number of things, an even lower number of things will be represented in the test set, resulting in a low level of serendipity.
Furthermore, this definition of relevance will exclude things that were not evaluated but would still be good recommendations for the user. Different algorithms' serendipity can still be compared within the same dataset, but this should be seen as a lower constraint on the system's overall serendipity.

- Novelty: Novelty is one of the fundamental aspects of a recommendation system since it increases efficacy and adds a new item to the suggestion list, contributing to superior accuracy. Novelty determines how unknown recommended items are recommended to a user. The capability of a recommender system to offer unique and surprising things that a user is unlikely to be aware of already is measured by novelty. It takes the suggested item's self-information to generate the mean self-information per the top N recommended list and averages it across all users. This metric can be extracted

from the feedback, which represent the discovery of new items. This is relevant as one of the goals of our approach was to provide novel, serendipitous items to the user. We propose a simple metric to evaluate this: the discovery index, to be computed by dividing the number of recommended items unknown to the user by the total number of provided recommendations (Equation (11)).

$$\text{Novelty} = \frac{\text{Number of recommended items unknown}}{\text{Total number of provided recommendations}} \quad (11)$$

Some recommender systems provide recommendations that are very accurate and have adequate coverage, yet are worthless for practical use [34]. One of the main indicators for studying recommender systems that assess a recommendation's "non-obviousness" is novelty. The term "novel recommendations" refers to suggestions of things that the user is unfamiliar with [16]. It is challenging to design metrics to assess novelty, since novelty is a measure of the degree to which recommendations deliver things that are both appealing to users and surprising to them. In reality, traditional ways of judging quality are fundamentally opposed to originality. To assess this measure, we will refer to the research of [35,36]. Reference [35] compared different novelty metrics in two larger datasets of movies (MoviesLens and Netflix), using different recommender methods: random, topN popularity recommendations; the probability of items being liked (PL); the probability of items being liked and dissimilarity (LD); and the probability of items being liked, dissimilarity, and satisfaction (LDS). The second article [36] introduced a formal framework for defining novelty and diversity measures, which unifies and generalizes various state-of-the-art metrics. The authors established a set of methodologies for measuring originality and variety based on the combination of ground components. The measures linked to the novelty (EPC and EPD) of a relevance-aware form of the content-based algorithm were used in their method.

5.3. Results

The results of all experiments are mentioned in this section. In addition, the collected results are discussed and utilized to compare those experiments' approaches.

Figure 6 presents the degree of novelty using content-based filtering. The reader can observe that the classic content-based recommender system algorithm has lower and minimal improvements when compared with other recommendation approaches illustrated in Figures 7 and 8.

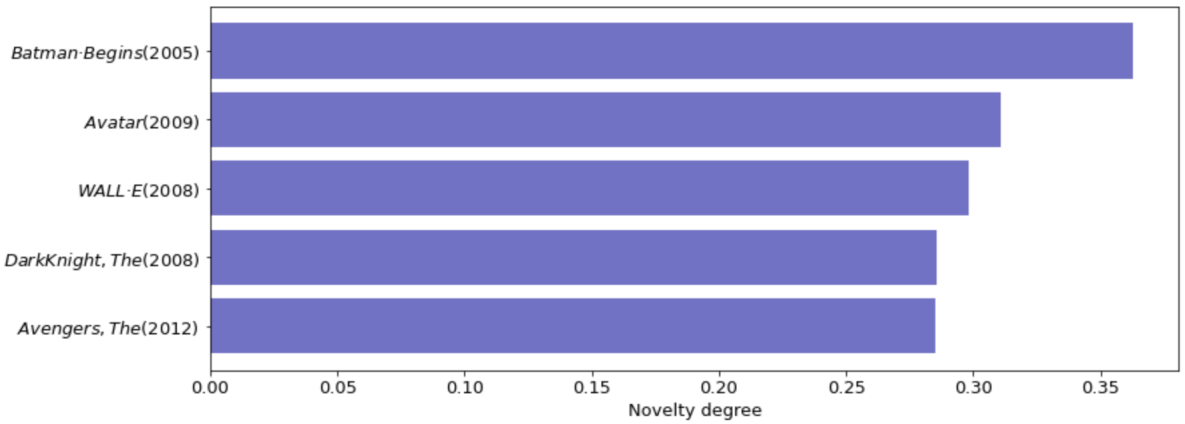

Figure 6. Recommendations of movies using content-based filtering.

Figure 7 presents the top five movie recommendations with the degree of novelty regarding the first initial population. The searched movie has Action, Adventure, Sci-Fi, Thriller, and IMAX as genres. All recommendation are performed based on the genre

criteria, for example the movie *An American Tail* is the highly similar one with a degree of 89.42%, which presents the similar Adventure genre and proposes the other new genres of Animation, Children, Comedy, which may be enjoyed by all users who like the searched movie.

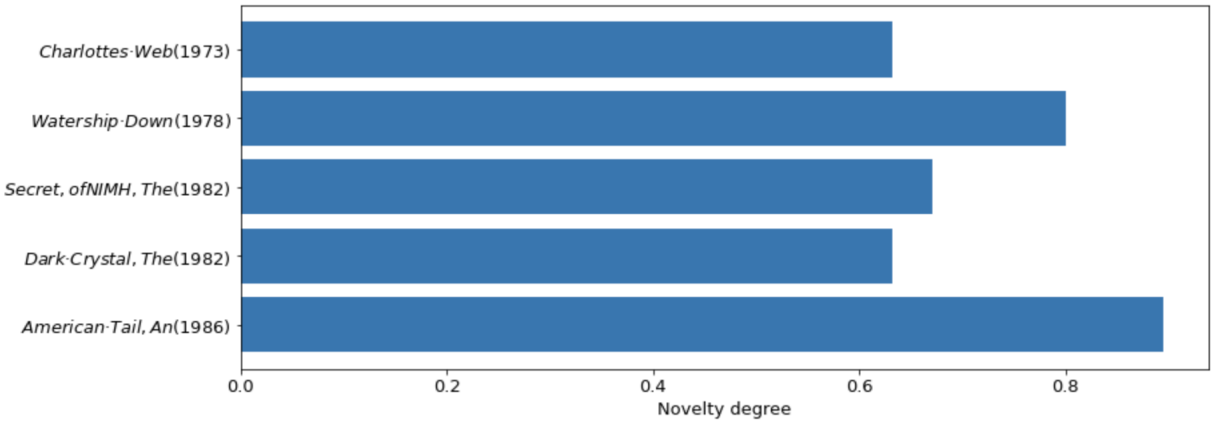

Figure 7. The recommendation of movies for initial population.

Figure 8 shows the top five movie recommendations for the last population. In this histogram we can compare between new and serendipitous items that are recommended to users. As we can see in Figure 8, both of the first movies have Adventure, Children, Comedy, and Fantasy as genres and have 89.44% as a degree of novelty because they offer three new genre that could be liked by the users.

The authors conclude that the performance of all recommendations improved from the initial to the last given results. The authors deduce that the higher the degree of novelty, the more novel and serendipitous the recommended element is, and the lower the degree of novelty, the more similar the recommended item is to the one sought.

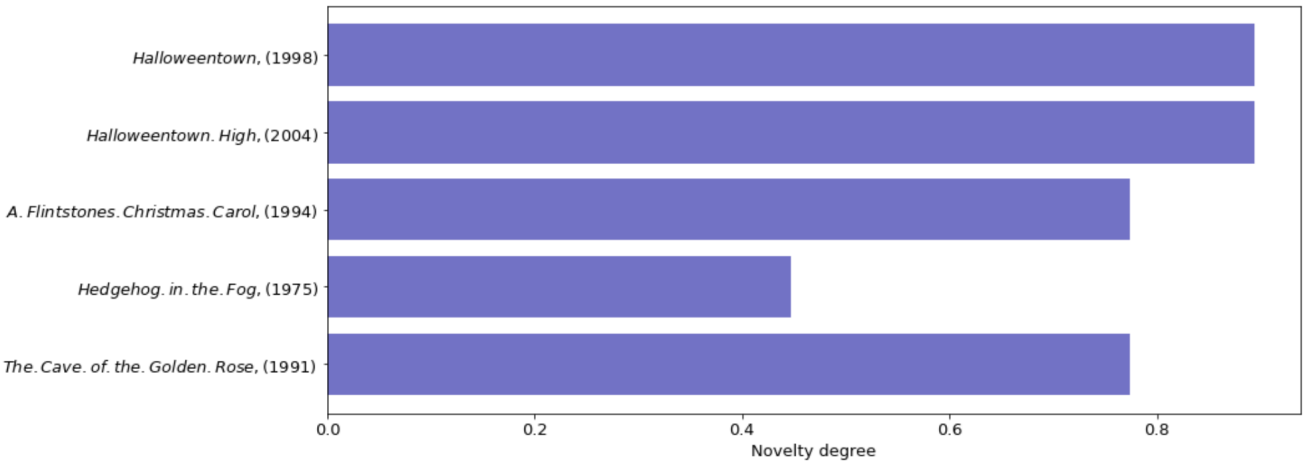

Figure 8. The recommendation of movies for last population.

The serendipity in Figures 7 and 8 can easily be seen on lines, with a high degree of novelty in the experience of receiving an unexpected and fortuitous item recommendation, therefore it is a way to diversify recommendations. While users rely on exploration and luck to find new items that they did not know they wanted, as an example a person may not know they likes watching *Halloweentown* until it is accidentally recommended to them. All recommendation are serendipitous, because they help the user to find a surprisingly interesting item that they might not otherwise have discovered. In conclusion, implementing operational serendipity techniques is an excellent way to enhance the capabilities of

content-based recommender systems in order to reduce the over-specialization problem by providing the user with unexpected ideas.

A sensitivity analysis was conducted to determine the suitable values that lead to the best GA performance. Table 4 presents the selected values of the parameters.

Table 4. The genetic algorithm parameters used in the experiment.

Parameter	Value
Number of generations	50
Population size	200
Mutation probability	5%
Crossover probability	5%
Top best individual percentage	5%

Table 5 describes the comparison between our proposed method and other recommender system approaches in terms of the diversity of the results, which is less if we use the classic content-based filtering and becomes higher when using the genetic algorithm that creates unexpected and fortuitous items in recommendation lists. Moreover, the effect of over-specialization is less in our suggested approach.

Table 6 shows the gathered novelty results. The reader can observe that $\boldsymbol{RRS_{GA}}$ shows remarkable improvements when compared with other recommendation approaches. The novelty of $\boldsymbol{RRS_{GA}}$ reaches its best case at Top 1 and Top 3, then begins to decrease. The obtained results demonstrate the superiority of the proposed method. Otherwise, on average, $\boldsymbol{RRS_{GA}}$ achieved 56% better novelty results than the content-based recommendation method.

Tables 7 and 8 present the precision and recall results of the recommendation methods. From these tables, the authors conclude that the performance of all recommendation methods with the increasing of the number of Top-N recommendation improved. This is because recall represents the percentage of the favorite recommended items out of all favorite items in the collection. Thus, increasing the number of recommended items leads to an increase in the probability of recommending interesting items for users.

Table 5. Comparison of our proposed approach with the recommender system approach.

Properties	Content-Based Filtering	Clustering	Proposed Approach—GA
Diversity of results	Less	Medium	High
Scalability	Low	Low	Good
Effect of over-specialization	High	Medium	Less

Table 6. Novelty results of the recommendation methods.

Method Recommendation	K = 1	K = 3	K = 5	K = 7	K = 9	K = 11
Content-based filtering	0.285	0.289	0.297	0.316	0.328	0.410
$\boldsymbol{RRS_{GA}}$	**0.845**	**0.740**	**0.726**	**0.6329**	**0.602**	**0.533**

Table 7. Comparison of precision results of both methods.

Method Recommendation	K = 1	K = 3	K = 5	K = 7	K = 9	K = 11
Using CB filtering	0.641	0.640	0.643	0.636	0.615	0.643
Using $\boldsymbol{RRS_{GA}}$	0.740	0.740	0.739	0.738	0.736	0.735

Table 8. Comparison of Recall results for both methods.

Method Recommendation	K = 1	K = 3	K = 5	K = 7	K = 9	K = 11
Using CB filtering	0.238	0.235	0.245	0.236	0.231	0.239
Using $\boldsymbol{RRS_{GA}}$	0.673	0.671	0.670	0.670	0.698	0.698

6. Discussion

The fundamental idea behind this study is to analyze potential recommendation lists rather than examining things and then building a recommendation list. As a result, $\boldsymbol{RRS_{GA}}$ looks for a suggestion list that fits three key features:

1. The suggested items are semantically correlated with the searched item.
2. The recommended goods represent a variety of what the user wants.
3. Unexpected and fortuitous items should be recommended to users.

To find a list that meets those features, $\boldsymbol{RRS_{GA}}$ adopts the revolutionary algorithms to have diversification in the recommendation list. The authors conclude that the size of the dataset and the number of recommended items impact $\boldsymbol{RRS_{GA}}$'s performance (i.e., size of the individual). Thus, the Top N should be chosen carefully and experimentally with a feasible dataset size to achieve the best possible performance. The results demonstrate $\boldsymbol{RRS_{GA}}$'s extraordinary progress in terms of prediction accuracy and suggestion quality.

The key obstacle affecting content-based filtering is over-specialization. As a result, $\boldsymbol{RRS_{GA}}$ intends to address the over-specialization issue in order to increase suggestion quality and RS accuracy. $\boldsymbol{RRS_{GA}}$ was tested on the MovieLens dataset. The improvements percentage achieved by $\boldsymbol{RRS_{GA}}$ when using Movielens was higher than those achieved by more classical methods.

Regarding the over-specialization issue, the $\boldsymbol{RRS_{GA}}$ outperforms the other RSs in terms of novelty, suggestion quality, and number of serendipitous items suggested to the user. Overall, $\boldsymbol{RRS_{GA}}$ made a significant improvement with respect to the recommendation quality. This demonstrates its effectiveness in alleviating the over-specialization problem faced by content-based filtering.

7. Distinctions

This section compares the suggested recommendation technique $\boldsymbol{RRS_{GA}}$ with existing recommendation systems (i.e., genetic-based and content-based). Instead of identifying familiar things to build the recommendation list, the suggested approach utilizes the GA to discover a suitable recommendation list that incorporates novel items. $\boldsymbol{RRS_{GA}}$ uses the GA to select the things on the list that best match the user's interests and are new. As a result, each individual in the population represents a potential suggestion list to the user. Based on the high score, the GA chooses one individual to suggest its genes to the user. To our best knowledge, no work in the recommender system research field has been identified in the literature that focuses on overcoming the over-specialization problem and suggesting new things to the user using genetic algorithms. The majority of works that deal with the over-specialization issue and the challenge of recommending new and unexpected items to the user do not use genetic algorithms in the context of information filtering. Table 9 shows the comparison of our proposed approach with other techniques that try to solve the over-specialization issue.

Table 9. Comparisons between our proposed approach and other approaches for mitigating the over-specialization problem.

Ref.	Handled Issues	Goal	Advantages	Limitations
[36]	Novelty and diversity	It focuses on diversity as an important feature of the recommendations. Diversity indicates how distinct the recommendations are when compared with each other.	A high potential for generalization and unification. Two novel features in novelty and diversity measurement arise from our study: rank sensitivity and relevance awareness.	A lack of exploration of other possibilities such as groups of user profiles, browsed items over an interactive session, or items recommended in the past.
[37]	Diversity	It introduces and explores a number of item-ranking techniques that can generate recommendations that have substantially higher aggregate diversity across all users while maintaining comparable levels of recommendation accuracy.	It provides significant improvements in recommendation diversity with only a small amount of accuracy loss.	Less recommendation quality and utility aspects.
[38]	Diversity	It proposes a hybrid recommendation algorithm based on communities of interest and trustworthy neighbors. The test results show that through modification of the diversity factor.	It integrates a user-preference-matching algorithm based on communities of interests and a diverse information recommendation algorithm based on trustworthy neighbors.	Very low accuracy on the diversity recommendation list.
[39]	Diversity and precision	It introduces a probabilistic structure to resolve the diversity–accuracy dilemma in recommender systems. It proposes a hybrid model with adjustable levels of diversity and precision. The proposed recommendation model consists of two models: one for maximization of the accuracy and the other for specification of the recommendation list to the tastes of users.	It introduces a probabilistic model, which has full control of the level of precision, novelty, and diversity.	-
Our Proposed approach	Diversity, novelty, and serendipity	It tackles the over-specialization problem in content-based recommender systems and innovates new items to the user. We are particularly interested in recommending surprising and beneficial items that may be enjoyed by the users.	Getting an unlikely and serendipitous article recommendation. It is a way to diversify recommendations, the user discovers new articles that they did not know they needed. Removes the limited content in content-based filtering.	Exploring only one semantic feature.

8. Conclusions and Future Work

Due to the presence of information overload, recommender systems have become increasingly crucial. We attempted to explore a new approach to tackle the over-specialization problem in content-based recommender systems and innovate new items for the user. The $\boldsymbol{RRS_{GA}}$ genetic algorithm was used to perform content-based filtering in this study. $\boldsymbol{RRS_{GA}}$ uses a genetic algorithm technique to make recommendations to the user. The main idea of this system is to search for a list that contains new items that highly correlate with user preferences and have a high probability (the proposed fitness function). $\boldsymbol{RRS_{GA}}$ searches for the correct recommendation list rather than the best items to form it. As a result, the initial population consists of a group of individuals, representing a collection of random possible suggestion lists. $\boldsymbol{RRS_{GA}}$ addresses the over-specialization problem in a novel way by considering each individual in the population as a potential recommendation list. Consequently, the suggested approach does not use a similarity metric to generate the recommendation list. Instead, it uses the similarity metric to test a possible recommendation list. The MovieLens dataset was used to test $\boldsymbol{RRS_{GA}}$ with the limited content issue caused by content-based filtering. In terms of prediction accuracy and suggestion efficiency, the proposed RS has been compared to alternative techniques. The collected results demonstrate that $\boldsymbol{RRS_{GA}}$ outperforms other techniques. This research has two drawbacks that could be discussed in future studies. This study tested the system on small and medium-sized datasets, which is not possible in a realistic situation. The use of only the item's genre function as a semantic feature results in the second limitation. In future research, more features should be explored and tested using the Boruta algorithm to maintain only the important features and have a satisfactory recommendation quality.

Author Contributions: Conceptualization, O.S.; Formal analysis, O.S.; Investigation, O.S.; Methodology, O.S.; Project administration, S.K. and O.B.; Resources, O.S.; Software, O.S.; Supervision, S.K. and O.B.; Validation, O.S., S.K. and O.B.; Visualization, O.S.; Writing—original draft, O.S.; Writing—review and editing, O.S. All authors have read and agreed to the published version of the manuscrip.

Funding: This research received no external funding.

Data Availability Statement: Publicly available dataset (Available online https://grouplens.org/datasets/movielens/, accessed on 2 March 2021) were used in this study, papers associated with these resources are properly cited.

Acknowledgments: The work presented in this paper was not supported by any institution.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Oumaima, S.; Soulaimane, K.; Omar, B. Latest Trends in Recommender Systems Applied in the Medical Domain: A Systematic Review. In Proceedings of the 3rd International Conference on Networking, Information Systems & Security, Shanghai, China, 23–25 April 2020.
2. Magdani, K.; Pasi, N.; Kackar, S.; Wakure, G. Music Recommendation in Artificial Intelligence Using Genetic Algorithm. 2016. Available online: http://ijarcet.org/wp-content/uploads/IJARCET-VOL-5-ISSUE-4-1039-1041.pdf (accessed on 2 May 2021).
3. Kim, H.-T.; Kim, E.; Lee, J.-H.; Ahn, C.W. A recommender system based on genetic algorithm for music data. In Proceedings of the 2010 2nd International Conference on Computer Engineering and Technology, Chengdu, China, 16–18 April 2010, Volume 6, pp. 414–417.
4. Soares, M.; Viana, P. Tuning metadata for better movie content-based recommendation systems. *Multimed. Tools Appl.* **2015**, *74*, 7015–7036. [CrossRef]
5. Reddy, S.R.S.; Nalluri, S.; Kunisetti, S.; Ashok, S.; Venkatesh, B. Content-Based Movie Recommendation System Using Genre Correlation. In *Smart Intelligent Computing and Applications*; Satapathy, S.C., Bhateja, V., Das, S., Eds.; Springer: Singapore, 2019; pp. 391–397.
6. Oumaima, S.; Soulaimane, K.; Omar, B. Artificial Intelligence in Predicting the Spread of Coronavirus to Ensure Healthy Living for All Age Groups. In *Emerging Trends in ICT for Sustainable Development*; Ahmed, M.B., Mellouli, S., Braganca, L., Abdelhakim, B.A., Bernadetta, K.A., Eds.; Springer International Publishing: Cham, Germany, 2021; pp. 11–18.
7. Stitini, O.; Kaloun, S.; Bencharef, O. The Recommendation of a Practical Guide for Doctoral Students Using Recommendation System Algorithms in the Education Field. In *Innovations in Smart Cities Applications*; Ahmed, M.B., Karaș, I.R., Santos, D., Sergeyeva, O., Boudhir, A.A., Eds.; Springer International Publishing: Cham, Germany, 2021; Volume 4, pp. 240–254.
8. Chen, H.-W.; Wu, Y.-L.; Hor, M.-K.; Tang, C.-Y. Fully content-based movie recommender system with feature extraction using neural network. In Proceedings of the 2017 International Conference on Machine Learning and Cybernetics (ICMLC), Ningbo, China, 9–12 July 2017; Volume 2, pp. 504–509.
9. Verma, A.; Virk, H. A Hybrid Recommender System using Genetic Algorithm and kNN Approach. *Int. J. Comput. Sci. Technol.* **2015**, *6*, 132–134.
10. Golbeck, J. Generating Predictive Movie Recommendations from Trust in Social Networks. In *Trust Management*; Stølen, K., Winsborough, W.H., Martinelli, F., Massacci, F., Eds.; Springer: Berlin/Heidelberg, Germany, 2006; pp. 93–104.
11. Adamopoulos, P.; Tuzhilin, A. On Over-Specialization and Concentration Bias of Recommendations: Probabilistic Neighborhood Selection in Collaborative Filtering Systems. In Proceedings of the 8th ACM Conference on Recommender Systems, Silicon Valley, CA, USA, 6–10 October 2014; pp. 153–160.
12. Beasley, D.; Bull, D.R.; Martin, R.R. *An Overview of Genetic Algorithms: Part 1, Fundamentals*; Inter-University Committee on Computing: Tel Aviv, Israel, 1993.
13. Alhijawi, B.; Kilani, Y. Using genetic algorithms for measuring the similarity values between users in collaborative filtering recommender systems. In Proceedings of the 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS), Okayama, Japan, 26–29 June 2016; pp. 1–6.
14. Alhijawi, B.; Kilani, Y. A collaborative filtering recommender system using genetic algorithm. *Inf. Process. Manag.* **2020**, *57*, 102310. [CrossRef]
15. Omar, B.; Zineb, B.; Cortés Jofré, A.; González Cortés, D. A Comparative Study of Machine Learning Algorithms for Financial Data Prediction. In Proceedings of the 2018 International Symposium on Advanced Electrical and Communication Technologies (ISAECT), Rabat, Morocco, 21–23 November 2018; pp. 1–5.
16. Lops, P.; de Gemmis, M.; Semeraro, G. Content-based Recommender Systems: State of the Art and Trends. In *Recommender Systems Handbook*; Ricci, F., Rokach, L., Shapira, B., Kantor, P.B., Eds.; Springer: Boston, MA, USA, 2011; pp. 73–105.
17. Heiss-Czedik, D. An Introduction to Genetic Algorithms. *Artif. Life* **1997**, *3*, 63–65. [CrossRef]
18. Srinivas, M.; Patnaik, L. Genetic algorithms: A survey. *Computer* **1994**, *27*, 17–26. [CrossRef]

19. Introduction to Genetic Algorithm and Python Implementation for Function Optimization. Available online: https://towardsdatascience.com/introduction-to-genetic-algorithm-and-python-implementation-for-function-optimization-fd36bad58277 (accessed on 18 May 2021).
20. Mühlenbein, H. How Genetic Algorithms Really Work: Mutation and Hillclimbing. In Proceedings of the Parallel Problem Solving from Nature 2, PPSN-II, Brussels, Belgium, 28–30 September 1992; pp. 1–12
21. Hamada, M.; Ometere, A.L.; Bridget, O.N.; Hassan, M.; Ilu, S.Y. A Fuzzy-Based Approach and Adaptive Genetic Algorithm in Multi-Criteria Recommender Systems. *Adv. Sci. Technol. Eng. Syst. J.* **2019**, *4*, 449–457. [CrossRef]
22. How to Define a Fitness Function in a Genetic Algorithm? Available online: https://towardsdatascience.com/how-to-define-a-fitness-function-in-a-genetic-algorithm-be572b9ea3b4 (accessed on 18 May 2021).
23. Alhijawi, B. The Use of the Genetic Algorithms in the Recommender Systems. *Gentic Algorithm* **2017**, *10*, 10–62.
24. Mohammadpour, T.; Bidgoli, A.M.; Enayatifar, R.; Javadi, H.H.S. Efficient clustering in collaborative filtering recommender system: Hybrid method based on genetic algorithm and gravitational emulation local search algorithm. *Genomics* **2019**, *111*, 1902–1912. [CrossRef] [PubMed]
25. Hassan, M.; Hamada, M. Genetic Algorithm Approaches for Improving Prediction Accuracy of Multi-criteria Recommender Systems. *Int. J. Comput. Intell. Syst.* **2018**, *11*, 146–162. [CrossRef]
26. Hwang, C.-S.; Su, Y.-C.; Tseng, K.-C. Using Genetic Algorithms for Personalized Recommendation. In *Computational Collective Intelligence. Technologies and Applications*; Pan, J.-S., Chen, S.-M., Nguyen, N.T., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; pp. 104–112.
27. Hasan, M.J.; Kim, J.-M. Fault Detection of a Spherical Tank Using a Genetic Algorithm-Based Hybrid Feature Pool and k-Nearest Neighbor Algorithm. *Energies* **2019**, *12*, 991. [CrossRef]
28. Marung, U.; Theera-Umpon, N.; Auephanwiriyakul, S. Top-N Recommender Systems Using Genetic Algorithm-Based Visual-Clustering Methods. *Symmetry* **2016**, *8*, 54. [CrossRef]
29. Chawla, S. Web Page Recommender System Using Hybrid of Genetic Algorithm and Trust for Personalized Web Search. *J. Inf. Technol. Res.* **2018**, *11*, 110–127. [CrossRef]
30. Rana, P.; Jain, N.; Mittal, U. *An Introduction to Basic Concepts on Recommender Systems*; Scrivener Publishing LLC: Beverly, MA, USA, 2020.
31. MovieLens Dataset. Available online: http://grouplens.org/datasets/movielens/ (accessed on 2 March 2021).
32. Peralta, V. Extraction and integration of movielens and imdb data. *Gentic Algorithm* **2007**, *17*, 1–17.
33. Shani, G.; Gunawardana, A. Evaluating Recommendation Systems. In *Recommender Systems Handbook*; Ricci, F., Rokach, L., Shapira, B., Kantor, P.B., Eds.; Springer: Boston, MA, USA, 2011; pp. 257–297.
34. Schafer, J.; Ben, F.D.; Herlocker, J.; Sen, S. Collaborative Filtering Recommender Systems. In *The Adaptive Web: Methods and Strategies of Web Personalization*; Brusilovsky, P., Kobsa, A., Nejdl, W., Eds.; Springer: Berlin/Heidelberg, Germany, 2007; pp. 291–324.
35. Zhang, L. The Definition of Novelty in Recommendation System. *J. Eng. Sci. Technol. Rev.* **2013**, *6*, 141–145. [CrossRef]
36. Vargas, S.; Castells, P. Rank and relevance in novelty and diversity metrics for recommender systems. In Proceedings of the RecSys'11, Chicago, IL, USA, 23–27 October 2011.
37. Adomavicius, G.; Kwon, Y. Improving Aggregate Recommendation Diversity Using Ranking-Based Techniques. *IEEE Trans. Knowl. Data Eng.* **2012**, *24*, 896–911. [CrossRef]
38. Liu, Q. Accurate and Diverse Recommendations via Integrated Communities of Interest and Trustable Neighbors. In Proceedings of the International Conference on Management of e-Commerce and e-Government, Shanghai, China, 31 October–2 November 2014; Volume 24, pp. 132–137.
39. Javari, A.; Jalili, M. A probabilistic model to resolve diversity–accuracy challenge of recommendation systems. *Knowl. Inf. Syst.* **2014**, *44*, 609–627. [CrossRef]