University Kasdi Merbah OUARGLA ALGERIA

Faculty of New Technologies of Information and Communication

Department of Computer Science and Information Technology

Domain: Mathematics and Computer science

Track: Computer science

Specialty: Fundamental

THESIS OF ACADEMIC MASTER

Presented by: Djidour Salma

Titled:

Multi-criteria Collaborative recommendation system with Self- Attention : study of impact of self-Attention in Prediction Phase

Publicly discussed on:

19/06/2022

Committee Members:

Mr.Belhadj Mourad	President	UKM Ouargla
Ms.Ameur Khadidja	Supervisor	UKM Ouargla
Mr.Said Bachir	Examiner	UKM Ouargla

Academic Year: 2021 / 2022

Acknowledgment

In the name of Allah, the Most Gracious and the Most Merciful. All praises to Allah and His blessing for the completion of this thesis.

This paper would not have been possible without the remarkable support of my supervisor, Dr. Ameur Khadidja, who provided us with invaluable advice and motivation, which kept this work on course from my first task to the final draft of this paper. Her knowledge, enthusiasm, and exacting attention to detail have been an inspiration that contributed tremendously to the successful completion of this work.

I would also like to express my special thanks and gratitude to my parents and colleagues at the University, who have helped me avoid numerous mistakes by providing helpful feedback and critiques on this research, allowing me to complete this project in the allotted time.

I am also grateful for their encouragement, patience, and support during the development of this work. The generosity and expertise of them all have improved this study in innumerable ways. Their willingness to generously give their time has been very much appreciated.

Abstract

In recent years, Recommendation systems (RS) has achieved great success and made remarkable progress in solving the information overload problem. There are many techniques used in the recommendation systems, collaborative filtering (CF) is one of the most used techniques. Several researches were proposed in the field of recommendation systems based on CF . Although existing models with single criteria have shown decent recommendation performance, they still suffer from challenges such as accuracy, data sparsity and cold start problem. However, multi-criteria predictions have been proved to be more accurate.

Lately, Self-Attention has achieved impressive results in several domains such as translation, voice, image recognition and computer vision. It recently gained massive interest in recommender systems, specially in sequential recommendation. However, through our search on this mechanism, we didn't encounter any study of using Self-Attention in multi-criteria collaborative recommendation systems.

In this work, we propose a multi-criteria collaborative filtering model in prediction phase based on Self-Attention mechanism, in order to study the impact of this mechanism in the prediction phase of recommendation systems. We conducted several experiments on real-world datasets. The obtained results of those experiments showed that the Self-Attention mechanism reduced sparsity of the datasets, and also, enhanced the prediction accuracy of our proposed model, and it outperformed the state of the art model.

Therefore, this study proves the success of employing multi-criteria and Self-Attention in the prediction of recommendation systems.

Key words: Recommendations Systems, Multi-criteria, Self-Attention, Collaborative Filtering.

ملخص

في السنوات الأخيرة ، حققت أنظمة التوصيات نجاحًا كبيرًا وأحرزت تقدمًا ملحوظًا في حل مشكلة زيادة المعلومات. هناك العديد من التقنيات المستخدمة في أنظمة التوصية ، والتصفية التعاونية هي واحدة من أكثر التقنيات استخدامًا. تم اقتراح العديد من الأبحاث في مجال أنظمة التوصية القائمة على التصفية التعاونية. على الرغم من أن النماذج الحالية ذات المعايير الفردية أظهرت أداء توصية لائقًا ، إلا أنها لا تزال تعاني من تحديات مثل الدقة وتناثر البيانات ومشكلة البداية الباردة. ومع ذلك ، فقد ثبت أن التنبؤات متعددة المعايير أكثر دقة.

في الآونة الأخيرة ، حقق الإهتمام الذاتي نتائج مبهرة في العديد من المجالات مثل الترجمة و الصوت والتعرف على الصور ورؤية الكمبيوتر. اكتسب مؤخرًا اهتمامًا كبيرًا بأنظمة التوصية ، خاصة في التوصية المتسلسلة. ومع ذلك ، من خلال بحثنا عن هذه الآلية ، لم نواجه أي دراسة لاستخدام الإهتمام الذاتي في أنظمة التوصية التعاونية متعددة المعايير.

في هذا العمل ، نقترح نموذج ترشيح تعاوني متعدد المعايير في مرحلة التنبؤ يعتمد على آلية الإهتمام الذاتي ، لدراسة تأثير هذه الآلية في مرحلة التنبؤ لأنظمة التوصية. لقد أجرينا العديد من التجارب على مجموعات البيانات في العالم الحقيقي. أظهرت النتائج التي تم الحصول عليها من تلك التجارب أن آلية الإهتمام الذاتي قللت من تباين مجموعات البيانات ، وعززت أيضًا دقة التنبؤ لنموذجنا المقترح ، وتفوقت على أحدث طراز.

لذلك أثبتت هذه الدراسة نجاح توظيف معايير متعددة والإهتمام الذاتي في التنبؤ بأنظمة التوصية.

الكلمات المفتاحية : نظام التوصيات, متعدد المعايير, الإهتمام الذاتي, التصفية التعاونية.

Résumé

Ces dernières années, les systèmes de recommandation (RS) ont connu un grand succès et ont fait des progrès remarquables dans la résolution du problème de la surcharge d'informations. Il existe de nombreuses techniques utilisées dans les systèmes de recommandation, le filtrage collaboratif (CF) est l'une des techniques les plus utilisées. Plusieurs recherches ont été proposées dans le domaine des systèmes de recommandation basés sur le CF . Bien que les modèles existants avec un seul critère aient montré des performances de recommandation décentes, ils souffrent toujours de défis tels que la précision, la parcimonie (la rareté des données) *sparsity* et le problème de démarrage à froid *cold start*. Cependant, les prédictions multicritères se sont avérées plus précises.

Dernièrement, Self-Attention a obtenu des résultats impressionnants dans plusieurs domaines tels que la traduction, la voix, la reconnaissance d'images et la vision par ordinateur. Il a récemment suscité un intérêt massif dans les systèmes de recommandation, en particulier dans la recommandation séquentielle. Cependant, à travers nos recherches sur ce mécanisme, nous n'avons rencontré aucune étude sur l'utilisation de l'attention à soi *Self-Attention* dans les systèmes de recommandation collaboratifs multicritères.

Dans ce travail, nous proposons un modèle de filtrage collaboratif multicritère en phase de prédiction basé sur le mécanisme attention à soi *Self-Attention*, pour étudier l'impact de ce mécanisme en phase de prédiction des systèmes de recommandation. Nous avons mené plusieurs expériences sur des ensembles de données du monde réel. Les résultats obtenus de ces expériences ont montré que le mécanisme attention à soi *Self-Attention* réduisait la rareté des ensembles de données et améliorait également la précision de prédiction de notre modèle proposé, et il surpasse le modèle de l'état de l'art.

Par conséquent, cette étude prouve le succès de l'utilisation de multicritères et d' attention à soi *Self-Attention* dans la prédiction des systèmes de recommandation.

Mots clés: Système de recommandations, Multi-critères, Attention à soi, Filtrage Collaboratif.

Contents:

List of Figures:

List of Tables:

List of Equations:

List of Abbreviations:

RSs: Recommendation Systems.

MCRSs: Multi-Criteria Recommendation Systems.

DL: Deep Learning.

CF: Collaborative Filtering.

CBF: Content Based Filtering.

CNN: Convolution Neural Network.

RNN: Recurrent Neural Network.

DNNs: Deep Neural Networks.

KBF: Knowledge-Based Filtering.

MF: Matrix Factorization.

SVD: Singular Value Decomposition.

MSVD: Multilinear Singular Value Decomposition.

ReLU: Rectified Linear Unit.

MAE: Mean Absolute Error.

MSE: Mean Square Error.

RMSE: Root Mean Square Error.

AICF: Attention-Based Item Collaborative Filtering.

GANCF: Gated and Attentive Neural Collaborative Filtering.

MDVAN: Multi-Criteria Dual-View Attention Network.

ANN: Artificial Neural Network.

NNs: Neural Networks.

General Introduction

1. Introduction:

The explosive growth in the amount of information is increasing far more than our ability to process it. This has increased the demand for recommendation systems (RSs). Recommendation systems (RSs) benefit both the user and the service producer, in revenues and user satisfaction. It is an information filtering technique that reduces the data overload problem, and aims to enhance the user experience by personalising item recommendation, and predicting a set of items that users may be interested in the most.

One of the most used methods in recommendation systems (RSs) is collaborative filtering (CF). It is basically an algorithm for matching users with similar interests in order to make recommendations. Various RS Collaborative filtering algorithms have been proposed over the years. And based on G. Adomavicius and A. Tuzhilin [1], we can identify those algorithms into two main classes used in the prediction phase: memory-Based method and Model-Based method. The memory-based method mainly analyses the user’s neighbourhood preferences from the user-item rating matrix. While the model-based method predicts the user’s rating of unseen items by developing models that supervise the way users rate items.

Usually, in most recommender systems, the utility function used to measure the suitability of recommending an item to a user is based on a single criteria value. It is an overall rating of an item generated by a user. But recently, it has been considered as limited because the recommendation may depend on more than one aspect. The use of multiple criteria allows a better representation of the user's preferences, which may lead to more effective and accurate recommendations.

However, the existing methods suffer from the lack of information, the datasets are extremely sparse and items can’t be reliably linked to users, causing a limitation in the recommendation’s effectiveness. This problem occurs due to insufficient or missing ratings of either the user or item or both in the dataset.

Recently, a new method has been introduced in Deep Learning (DL) which is Self-Attention. It was proposed in the paper “Attention is All You Need” [2], it is a mechanism that allows the inputs to interact with each other, and find out who they should pay more attention to. The outputs are aggregates of these interactions and attention scores. Through our search on this mechanism, we didn't encounter any study of using Self-Attention in multi-criteria collaborative recommendation systems. Most Self-Attention mechanism researches are made in the sequential recommendation. According to [3], a Sequential recommender works by exploiting a sequence of information from the user history, and using it to predict the possible next user-item interaction. Then it uses the Self-Attention mechanism to focus on items relevant to the next action. The researchers in this field [4] have found that using Self-Attention outperforms and is faster than traditional models CNN/RNN based approaches, and also enhanced the interpretability of recommendation algorithms. Due to this significant improvement, we are inspired to apply the Self-Attention mechanism in a multi-criteria collaborative recommender system.

Our main objective is to find out if it is possible to use the Self-Attention mechanism in the prediction phase and how we can achieve that, then study the impact of this mechanism in multi-criteria collaborative recommendation systems on the quality of the prediction, and if it is efficient to reduce the problem of data sparsity.

Our study is structured as follows: In the first chapter, we present a comprehensive review of the state of the art in recommender systems, identifying some basic notation about the collaborative filtering approach with its methods and limitations. We will also explain the multi-criteria recommendation systems, then introduce the Self-Attention mechanism along with its working process. We conclude this chapter with a brief discussion about the traditional models and formalise their problem that inspired us to propose a new model. This will lead us to the second chapter, where we will discuss our methodology and explain our proposed model. Next, we reach the third chapter, where we perform a series of experiments to demonstrate the efficiency of using Self-Attention, and discuss the obtained results of our model. Finally, we conclude this study by a brief recap of our search, and with an extensive overview on some issues and future challenges for further research in the field.

Chapter 01

State of The Art

Nowadays, there have been a vast flow of information due to the various services and platforms that the web provides such as marketing (Amazon), social media (Facebook), e-learning (Udemy), movies (Netflix) and videos streaming (Youtube). Despite the benefits of this data, it causes challenges for users to choose from a massive number of items.

Due to this information overload problem, it has rather become a necessity to use recommendation systems (RSs) that sense what the user may like and needs. It provides personalised suggestions, recommends items to users based on their preferences, and helps in the decision-making process. In recent years, recommendation systems has achieved great success and made remarkable progress, therefore, it has become a trending topic in the academic sector, and several researches were proposed in this field.

1. Recommendation System (RS):

Recommendation systems are considered as a decision making strategy for users. It is a data filtering tool that aims to reduce the information overload problem by collecting users information to predict items that are close and relevant to the user interest and preferences. Some examples of recommender systems include:

- Product recommendations on Amazon and other shopping sites.
- Movie and TV show recommendations on Netflix.
- Article recommendations on news sites.

An example of Netflix recommendation interface is shown in Figure 1.1.

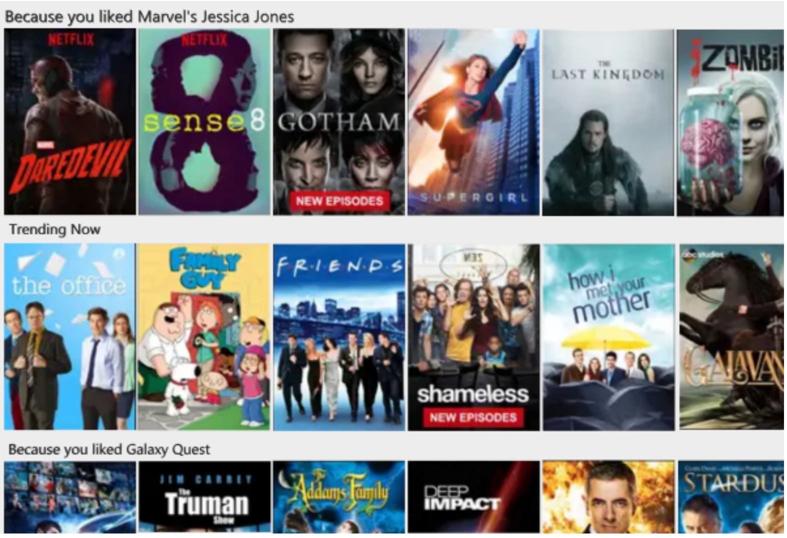


Figure 1.1: Example Of Netflix Recommendations For Movies.

Based on "Introduction to recommender systems handbook” [8], there are three main kinds of relationships in RS : “user-user”, “item-item” and “user-item”.

- The user-user recommendation is selecting a neighbourhood (group) of users who are similar to the target user and analysing their preferences to provide suggestions of unrated items that were already liked by the users in his neighbourhood.
- The item-item recommendation occurs when some items are similar in nature, either by appearance or description.
- The user-item recommendation occurs when some users have an attraction towards specific items, the recommendation system will build a ”user-item” relation.

Generally, a Recommendation System model consists of:

- Users set contains all the users.
- Items set contains all the items that can be recommended to the users.
- A utility function that calculates the suitability of a recommendation to user u ∈ Users and item i ∈ Items, which is generally declared as R: Users×Items→r_0, where r_0 is a positive integer within a specific range [5].

Recommendation Systems works through three main phases as shown in Figure 1.2. we demonstrate those phases as follows:[5]

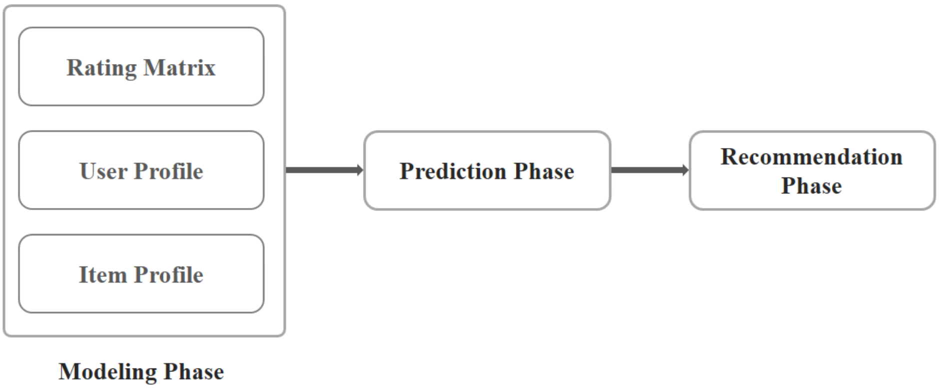


Figure 1.2: Recommendation Systems Phases based on [5]

- **Modelling Phase:** This phase prepares the data that will be used in the next two phases. There are three ways for that:

- Building a rating user-item matrix, filled with the ratings of the users for items.
- Building a user profile as a vector for each user that explains his preferences of an item.
- Building an item profile that contains the features of a specific item.

- **Prediction Phase:** This phase estimates the utility function R of Users x Items for a user on an item, based on known ratings given by users explicitly for items and/or transaction data implicitly that shows users' preferences such as user profiles, item content and purchase history such as numeric ratings. It calculates the predictions of ratings for the items to produce a numerical value R_{ui}, it aims to predict the rating of unseen item i ∈ Items for a user u ∈ Users, through a utility function depending on the information extracted during the previous phase.

- **Recommendation phase:** This phase in which the calculated predictions of the previous phase are used to recommend a set of top N-items with the highest predicted ratings, which means items that the user may like the most.

In the literature, recommendation systems (RSs) are usually categorised into four main categories: content based filtering (CBF), collaborative filtering (CF), knowledge based filtering (KBF) and hybrid [8]. Figure 1.3 shows the different recommendation filtering techniques. In our study, we will focus on the collaborative filtering approach since it is the most popular technique used in recommendation systems and it proves that it gives the most accurate results.

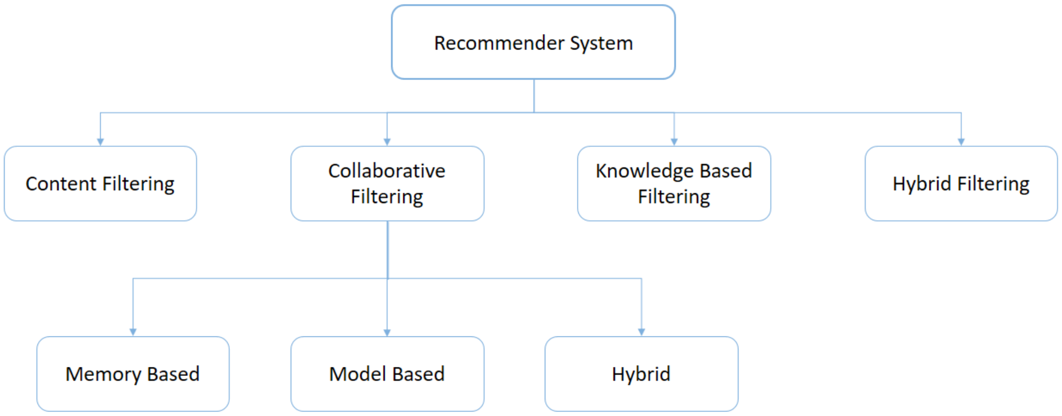


Figure 1.3: Recommendation Systems Approaches based on [6]

1.1. Collaborative Filtering Approach:

Based on "Recommendation systems: Principles, methods and evaluation" [7] and "Introduction to recommender systems handbook" [8], Collaborative filtering (CF) is the most prominent approach in recommender systems. It generates the recommendation for a user based on other users who have similar preferences as the targeted user. This approach according to [19, 20] is based on the assumption that people who matched with a user in the past will also match in the future. Which means that the predicted rating of a user u for a new item i is likely going to be similar to user v rating on this item, if u and v have similar ratings on other items in the past.

Collaborative approaches overcome some of the limitations of other approaches, because it does not rely on the service content in which it is applied nor user or items information, recommendations are made only with users ratings on items and that gives it an advantage to recommend any kind of items.

A Collaborative filtering system contains a list of Users = $[u_1, u_2, \ldots, u_n]$ and Items = $[i_1, i_2, \ldots, i_m]$, this constructs an $n \times m$ user-item matrix of ratings, where each set $r_{u,i}$ indicate the rating of a user u on an item i. It then matches users with relevant interest by calculating similarities between them to make recommendations by either predicting the rating for an active user u on item i or recommends a top-N list of items that will mostly interest user u [21]. An illustration of the general CF process is shown in Figure 1.4.

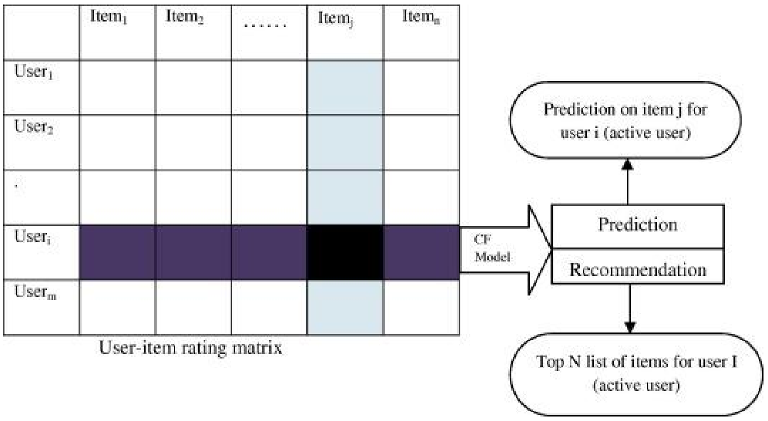


Figure 1.4: General Collaborative Filtering Process based on [7].

According to [8, 9, 10, 11], the CF approach can be classified into two main categories of memory-based and model-based, as shown in Figure 1.5 below.

1.1.1. Memory-based (heuristic-based) collaborative filtering:

Is a heuristic algorithm that uses the user-item matrix to identify users/items that have similar rating patterns. The ratings of these entities (users/items) are used to predict an unspecified item's rating for a certain user. Memory-based algorithms can be user-based or item-based. the past preferences (ratings) of similar users of a certain user u are employed in user-based CF, on the other hand, ratings of similar items to a certain item i are used in item-based CF [22]. In this approach, unlike Model-based there are no learning parameters involved, except the similarity which can be calculated using cosine similarity or Pearson Correlation Coefficient [18]. However, these methods are common in single-criteria, we cannot directly employ them in multi-criteria rating because they need an overall rating. So two main approaches can be used:

First approach: applying the previous methods between two users on each individual criteria separately. Then compute an aggregate similarity over all criteria using one of the following methods: Average similarity, Worst-case (smallest) similarity, Aggregate similarity.

Second approach: using multidimensional distance metrics such as: Manhattan distance, Euclidean distance, Chebyshev (maximal value) distance. The smaller is the distance between users the similar their rating values are.

1.1.2. Model-based collaborative filtering:

It adopts machine learning or data mining algorithms to build the CF predictive model that learns from the given data (specified ratings) to make a prediction about the unspecified values (unknown ratings). The model trains on the user-item interactions (ratings) and then predicts ratings of users for new items. Model-based CF algorithms include: Bayesian models, clustering models, decision trees, and singular value decomposition models (SVD), Aggregation function approach, Multilinear singular value decomposition (MSVD) approach, Matrix factorization (MF) etc... [8, 23].

Singular value decomposition (SVD) technique: is a well known technique adopted in CF approach. It has gained popularity in recommender systems because of its efficiency in improving recommendation performance accuracy. It is used on single criteria rating recommendation systems, as it identifies hidden features of users and items.

Multilinear singular value decomposition (MSVD) technique: is used in multi-dimensional datasets on multi-criteria ratings, where it learns to detect relationships between users and items, and uses them to predict criteria ratings to a target user. The MSVD technique uncovers relationships between users and items and then uses this information to predict a rating and compute the top-N recommendations list. This technique according to [8] improves the recommendation accuracy compared to the SVD model in single criteria.

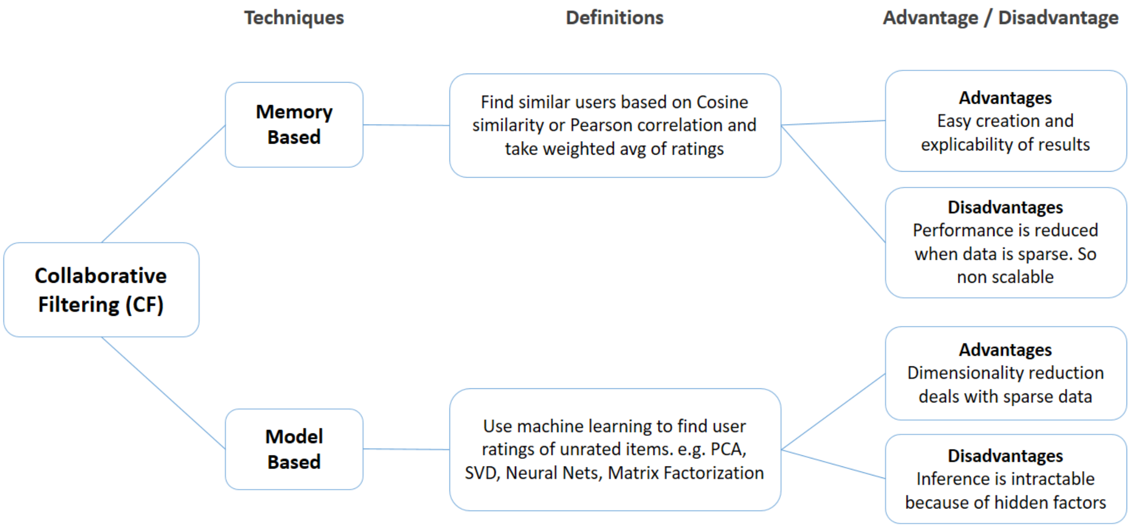

Techniques
Definitions
Advantage / Disadvantage
Memory Based
Find similar users based on Cosine similarity or Pearson correlation and take weighted avg of ratings
Advantages
Easy creation and explicability of results
Disadvantages
Performance is reduced when data is sparse. So non scalable
Collaborative Filtering (CF)
Advantages
Dimensionality reduction deals with sparse data
Model Based
Use machine learning to find user ratings of unrated items. e.g. PCA, SVD, Neural Nets, Matrix Factorization
Disadvantages
Inference is intractable because of hidden factors

Figure 1.5: Collaborative Filtering Classes based on [8]

Even though recommender systems provide effective ways to deal with the information overload problem, it also faces many challenges, including accuracy, sparsity, cold-start, and scalability.

1.2. Challenges of Recommendation System Collaborative Filtering:

The CF approach's performance depends on the availability of sufficient rating information. And despite this technique's success it still has some issues such as:

- **Sparsity Problem:** Refers to insufficient information of either the user or item or both in the dataset, that is, when only a small fraction of the items are rated by users and that is caused by the lack of users participation. Therefore, the recommendation system cannot be able to generate accurate predictions without sufficient information.
- **Cold Start Problem:** Is the case where we don't have enough information about the user (new user who has not rated any item) or the item (new items that have not been rated yet) in order to make a proper prediction. We can consider the cold start problem as a special case of sparsity since the database will contain empty values.
- **Scalability:** The number of users and items in the database is increasing rapidly. The recommendation system should have a quick response time to keep users satisfied. Large datasets and responding time create a challenge in creating an effective recommendation system.
- **Synonymy:** When we have similar items with different names or categories. It is difficult for the recommendation system to distinguish between those items, for example when we have kids outfits and kids clothes. That leads to poor recommendations.

Recommendation systems were first used on single-criteria, and through research in this field, a new concept was defined by extending the single-rating to a multi-criteria rating. The multi criteria recommendation systems (MCRSs) showed more accurate performance due to its complex presentation of the user’s interests.

2. Multi-Criteria Recommendation Systems :

Earlier, recommendation systems were used in two-dimensional datasets of Users and Items. They generally use single-criteria ratings that indicate the overall utility of an item by a user. The utility of user on item, is represented by a set r_0, where r_0 is equal to a positive integer within a specific range [5]. The utility function R is declared as [8]:

$$R: Users * Items \rightarrow r_0 \text{ (1.1)}$$

For instance, user1 rated a certain item (item3) 3 out of 5 with a single criteria rating, we can represent this by R(user1, item3) = 3. This rating indicates how much a user likes an item. But this method has been considered as limited, because the recommended item for a user may depend on more than one perspective that the user takes into consideration when choosing an item.

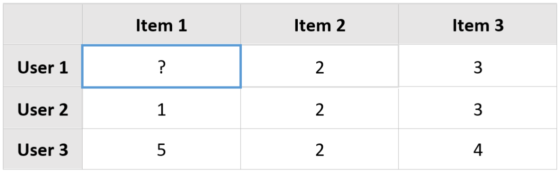
Therefore, a new concept that employs multiple criteria has emerged. The multi-criteria ratings provides additional information that improve the performance of the recommendation, it gives accurate predictions due to its complex representation of each user's preferences. Multi-criteria rating systems use a set of ratings for each individual criteria r_1,...,r_c (c is the number of criteria) given by a user for an item, while some datasets also include the overall rating r_0 (Figure 1.9). we can present the utility function of the multi-criteria recommendation with the overall rating or without it as follows [8]:

$$R: Users * Items \rightarrow r_0 * r_1 *... * r_c \text{ (1.2)}$$

or

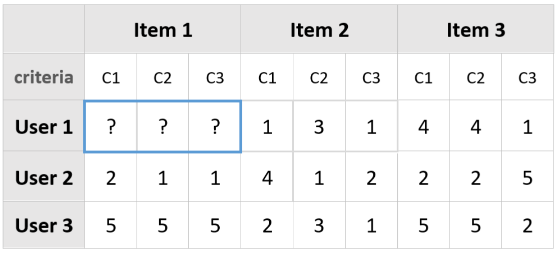
$$R: Users * Items \rightarrow r_1 *... * r_c \text{ (1.3)}$$

Using the most similar user to predict a user's rating for a certain item (assuming that the ratings have been predicted using any of the techniques discussed in the section above), for illustration, we are trying to predict the utility of item1 for user1. We can see that user1 and user2 have similar overall rating on the items that both of them have already seen and rated in figure 1.7, therefore, the rating of item1 for user1 would be predicted as 1 based on user2 rating. The benefits of using multi-criteria ratings is that the information are more specific therefor to predict an item to a certain user is more efficient to find a similar user in preferences, for example user1 and user2 have similar preferences in a single-rating (Figure 1.7), but in a multi-criteria rating (Figure 1.8) we can see that they have different preferences when it comes to item aspects, even though they had the same overall ratings. On the other hand, user1 and user3 have very similar ratings patterns, therefore user3 is more similar to user1 than user2. Thus, using the same technique which is finding the most similar user, but taking into consideration multi-criteria ratings, user1's overall rating for item1 would be predicted as 5, based on user3's ratings for that item.



	Item 1	Item 2	Item 3
User 1	?	2	3
User 2	1	2	3
User 3	5	2	4

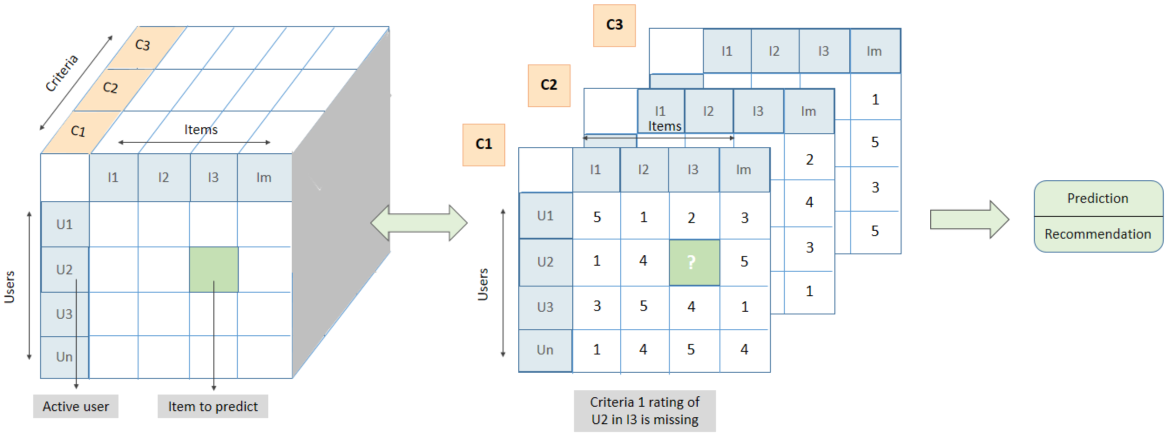
Figure 1.7: Overall Rating Recommendation Matrix.



	Item 1			Item 2			Item 3		
criteria	C1	C2	C3	C1	C2	C3	C1	C2	C3
User 1	?	?	?	1	3	1	4	4	1
User 2	2	1	1	4	1	2	2	2	5
User 3	5	5	5	2	3	1	5	5	2

Figure 1.8: Multi-Criteria Recommendation Matrix.

As a result, we can say that a single overall rating may not show the variation of users' preferences for different aspects of a certain item, whereas multi-criteria ratings may help the recommendation system in better understanding each user's preferences, which leads to providing users with more accurate recommendations.



	I1	I2	I3	Im
U1	5	1	2	3
U2	1	4	?	5
U3	3	5	4	1
Un	1	4	5	4

Figure 1.9: Multi-Criteria Recommendation.

Although using multi-criteria in CF approach has shown decent performance in recommendation systems, it lacks in addressing cold start and data sparsity issues. However, a new concept called Self-Attention has been introduced, that shows promising results when it comes to finding complex relationships through simple interactions among users and items.

3. Self-Attention:

Self-Attention or Intra-Attention was proposed in the "Attention is All You Need" [2] paper, by researchers at Google Research and Google Brain, due to challenges faced by encoder-decoder in dealing with long sequences. Self-Attention is a special case of the attention mechanism and unlike basic attention that learns representations with limited knowledge of the whole context, Self-Attention can capture the relationships between elements, regardless of their distance. It has only started to gain exposure due to its recent successful application on machine translation. It can replace RNN and CNN in sequence learning, achieving better accuracy with lower computation complexity. This mechanism works independently, without any recurrence or convolution (unlike basic attention). And instead of focusing attention on only a fixed user preference, it extracts information from the whole sequence allowing the inputs to interact with each other to find out who they should pay more attention to, taking into consideration the diversity of user preferences.

The building block of Self-Attention (figure 1.10) is scaled dot-product attention. The input of the attention module consists of a query (Q), key (K), and value (V) that takes the rating matrix. For the Self-Attention layers it is always Q = K = V. The query (Q) and key (K) are then dot-producted and scaled. These are softmaxed to obtain attention scores producing the final attention matrix. It's basically adding attention score to the ratings based on similarities. The Self-Attention formula is presented as follows:

$$Attention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_K}})V \quad (1.1)$$

According to [2], the dot product produces large values, pushing the softmax into extremely small gradients. Therefore, we scale the dot products by $\sqrt{d_K}$ to avoid vanishing gradients.

An illustration of the Self-Attention block is shown in Figure 1.10 below.

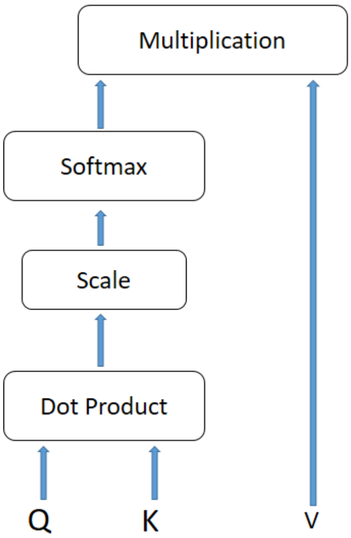
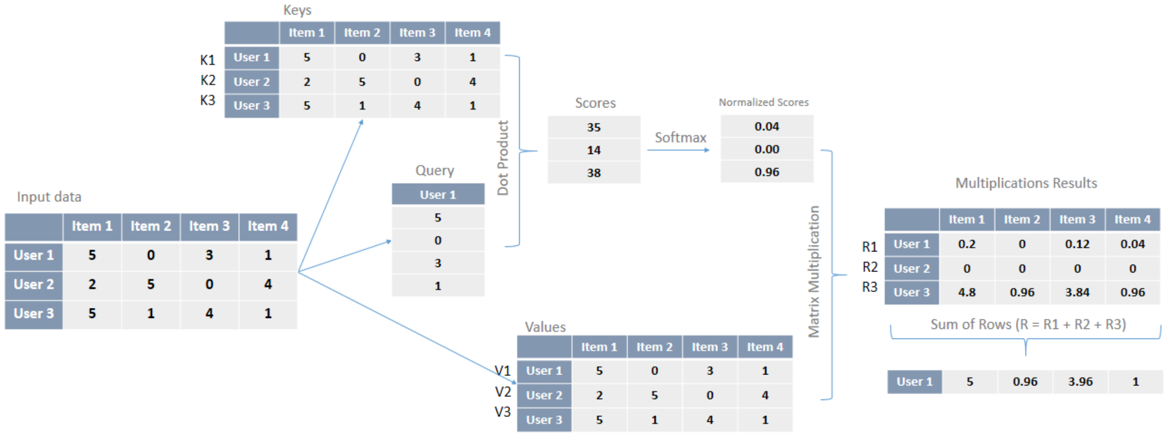


Figure 1.10: Illustration of the main idea of Self-Attention with key, query and value transformations.

Query (Q) with all Keys (K), are dot-producted and divided by $\sqrt{d_K}$, where the Keys take each user at a time, this multiplication will produce the scores vector. Then, the softmax function is applied to that vector to obtain the normalised scores that will be multiplied by Values (V), producing a matrix with values representing the effect amount of each user in each item that will be applied on the target user in Keys (based on similarity between them). Finally, the scores are then summed together producing the target user' new ratings.

To understand more how Self-Attention works, we present an example in Figure 1.11 below that illustrates different steps of the Self-Attention model.



Input data

	Item 1	Item 2	Item 3	Item 4
User 1	5	0	3	1
User 2	2	5	0	4
User 3	5	1	4	1

Keys

	Item 1	Item 2	Item 3	Item 4
User 1	5	0	3	1
User 2	2	5	0	4
User 3	5	1	4	1

Query

User 1
5
0
3
1

Scores

35
14
38

Normalized Scores

0.04
0.00
0.96

Values

	Item 1	Item 2	Item 3	Item 4
User 1	5	0	3	1
User 2	2	5	0	4
User 3	5	1	4	1

Multiplications Results

	Item 1	Item 2	Item 3	Item 4
User 1	0.2	0	0.12	0.04
User 2	0	0	0	0
User 3	4.8	0.96	3.84	0.96

User 1	5	0.96	3.96	1

Figure 1.11: Example of Self-Attention.

A comparison of the computational complexity for Self-Attention was done by [29], where n is the length of input, d is the dimension of input and k is the kernel size on convolution.

Layer Type	**Complexity**
Self-Attention	$O(1)$
Recurrent Neural Network	$O(n)$
Convolutional Neural Network	$O(log_k(n))$

Table 1.1: Comparison of complexity for Self-Attention, RNN, and CNN .

There are several studies that implement multi-criteria ratings and Self-Attention in recommendation systems. Therefore, in the next section we will briefly present some state of the art research that implements Self-Attention in recommendation systems and multi criteria in collaborative filtering.

4. Related Work:

Before we get into the core of creating our model, we need to review some studies that are made in these aspects: Multi-criteria Recommendation, Collaborative Filtering, and Self-Attention Mechanism.

There are some works conducted in Deep learning that use multi criteria, and have achieved impressive results in recommender systems predictions, such as the work by Nassar et al (January 2020) that suggests a novel multi-criteria collaborative filtering model based on deep learning. The model uses the users and items' features as inputs to the criteria ratings deep neural network, which predicts the criteria ratings. Those criteria ratings constitute the inputs to the overall rating deep neural network to predict the overall rating. It achieved good performance which proves that the use of multi-criteria and deep learning in collaborative filtering recommendation system is more accurate than single criteria.[16]

There is also the work of Moqa, Salem, et al (November 2021), that proposed a multi-criteria Dual-View Attention Network (MDVAN) to predict the multi-criteria ratings for items as they progress from coarse to multi-criteria fine attention level. This work shows that dual-view attention network in multi-criteria recommendation systems achieved great success.[17]

According to a paper called “Amazon.com Recommendations: Item-to-Item Collaborative Filtering” [12], Amazon currently uses item-to-item collaborative filtering, which scales to massive data sets and generates high-quality recommendations in real time. With item-to-item collaborative filtering, the recommendation algorithm would review the user’s recent purchase history, and for each purchase it pulls up a list of related items. Items that appear often across all the lists are possible items to recommend to the user. Then each of these items would have a weight depending on how related they were to the user’s previous purchases. This recommendation algorithm is effective for providing a personalised shopping experience for the user, and that helps Amazon increase the amount of their revenue.

There are some works that employed the Self-Attention mechanism, such as the Next Item Recommendation with Self-Attention paper, where authors proposed a novel Self-Attention sequential recommendation model called AttRec [13]. The model consists of a Self-Attention module, and a collaborative metric learning component. It uses Self-Attention to learn the user's short-term preferences (users' purchasing inclination in a relatively short period) from his recent actions. While the collaborative matrix is used to model user long-term preference (user' inherent purchasing bias and evolves slowly). The model incorporates both short and long term preferences to predict the next actions.

The attention-based item collaborative filtering model (AICF) has shown to have a higher recommendation accuracy. AICF adopts three layers of attention: the Dot-product Attention, Self-Attention, and Transformer model, to estimate the weights of historical items to represent the profile of users. AICF architecture has a three-layer attention, to assign dynamic weights to different historical interactive items to measure their importance. It uses the Dot-product attention to model the relationship between users and items more accurately, and accelerate the processing speed of recommendation systems, and improve the space utilisation rate. The Self-Attention maps the relationship between users and items to different subspaces, and expands the ability of the model to focus on items in different locations, it also improves the precision of the recommendation system. The Transformer makes the model more comprehensive to capture user preferences and improve the recommendation performance of the system. [15]

Attention mechanisms have been shown to be effective in various tasks such as machine translation. The attention technique is essentially an additional component to the original model RNN. Recently, a new method has been introduced by Attention is all you need paper called the Transformer [2], it has been efficient on machine translation tasks. It is designed to process sequential input data for natural language tasks. Similar to the Sequence-to-Sequence machine translation model, the Transformer model is also based on the encoder-decoder architecture. However, unlike RNNs, it does not necessarily process the input data in sequential order. The Transformer model relies entirely on the Self-Attention module to capture complex structures in sentences, and to retrieve relevant words for generating the next word. The Transformer model extracts features for each word using a Self-Attention mechanism to figure out how important all the other words in the sentence are to the aforementioned word. And no recurrent units are used to obtain these features, they are just weighted sums and activations, so they can be very parallelizable and efficient.

5. Discussion and Conclusion:

Through our search on the Self-Attention mechanism, we found that there are some works that employed multi-criteria and basic Attention mechanisms in collaborative filtering recommendation systems. Those works achieved significant performance in the ratings prediction. However, we didn't encounter any study that employs the Self-Attention mechanism in multi-criteria collaborative recommendation systems. Due to the significant performance of this mechanism in the sequential recommendation and single criteria recommendation, and based on the searches in the previous section, we are inspired to apply the Self-Attention mechanism in a multi-criteria collaborative recommender system.

This chapter contains a background of our research, starting from identifying the recommendation systems and collaborative filtering with its methods and the challenges in this field. Then, we explained the multi-criteria recommendation in detail. Moreover, this chapter provides an introduction to the Self-Attention mechanism, and some related work. We end this chapter by a brief discussion and a conclusion.

Chapter 02

Our Methodology

This chapter is a discussion of the methodology used in this work. Starting by explaining the techniques used including the Self-Attention mechanism and the Multi Head Attention. Then, present a description of our prediction models, followed by the steps of implementing the Self Attention and the Multi Head Attention in the models. At last, we end this chapter with a conclusion.

1. System Overview:

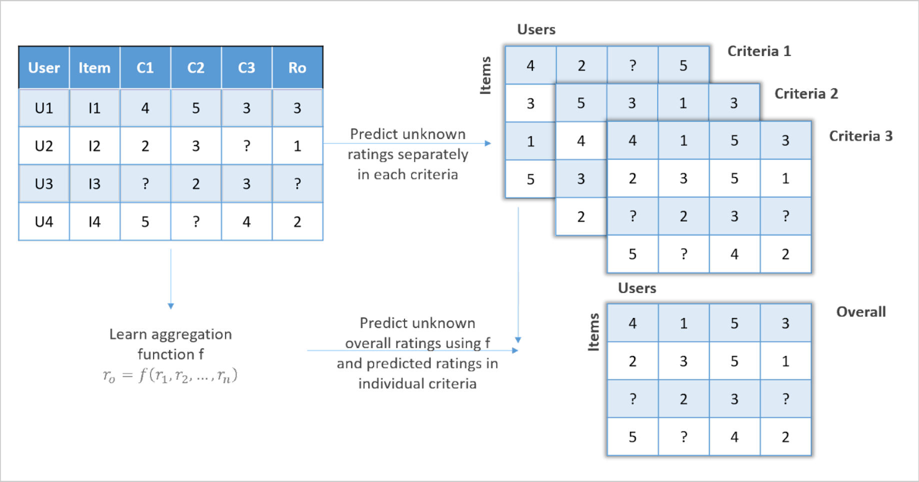
Our proposed models aim to apply the Self-Attention on a Multi-Criteria Collaborative Filtering recommender system for rating prediction. Using one of collaborative filtering technique approaches in the model-based for multi-criteria recommendation, the Multilinear singular value decomposition (MSVD) approach, to study the impact of Self-Attention in the prediction phase on a multi-criteria dataset. Each proposed model is conducted to predict the overall rating and the criteria ratings.

1.1. Data Modelling:

Our Collaborative filtering system uses the explicit interaction (user's previous ratings) to learn the relationship between user and item in order to make a prediction. Users and items are represented by their ids. the models are conducted in two approaches as illustrated in Figure 2.1 below:

Predict criteria ratings (MSVD): In this step, we will predict the unknown multi-criteria ratings for a user on an item, using the MSVD model. We used only user and item ids as features. Since the ids are a general feature representation, our model can be implemented to reduce the cold start problem. The model takes the users and items' ids and uses them as an input to the MSVD model, to predict the criteria ratings.

Predict overall rating (Aggregation): The overall rating can be estimated using an aggregation function of multi-criteria ratings. In this step, we aim to learn the relationship between the overall rating and the criteria ratings, using the criteria ratings as inputs to the aggregation function, to predict the overall rating.



User	Item	C1	C2	C3	Ro
U1	I1	4	5	3	3
U2	I2	2	3	?	1
U3	I3	?	2	3	?
U4	I4	5	?	4	2

Criteria 3

4	1	5	3
2	3	5	1
?	2	3	?
5	?	4	2

Overall

4	1	5	3
2	3	5	1
?	2	3	?
5	?	4	2

Users
Items
Criteria 1
4 2 ? 5
3
1
5
Criteria 2
5 3 1 3
4
3
2
Predict unknown ratings separately in each criteria
Learn aggregation function f
$r_o = f(r_1, r_2, \ldots, r_n)$
Predict unknown overall ratings using f and predicted ratings in individual criteria
Users
Items

Figure 2.1: Representation of the two prediction branches: MSVD and Aggregation based on [8].

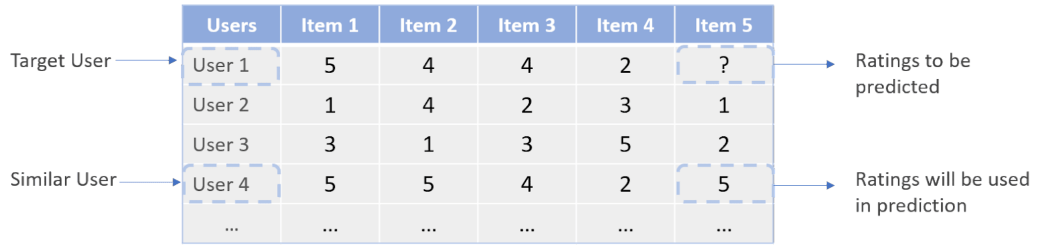
1.2. Our Proposed Models:

In this section, we will explain the proposed prediction models that we used to predict the overall ratings and the criteria ratings, starting from a comparative study between single criteria and multi-criteria prediction systems. Then we are going to implement the Self-Attention mechanism to the previous basic models, in order to study the impact of this mechanism on the performance of the prediction. Lastly, we will implement the Multi Head Attention to compare its performance with the Self-Attention models.

1.2.1. Constructing Basic Models:

We conducted a comparative study in the single criteria and multi criteria prediction system, to compare between their performance and effectiveness. We start by how we conducted the single criteria model.

Model 1: Single criteria using SVD approach.
In single criteria we have only one rating which is the overall rating, we used the available rating to train our Singular Value Decomposition (SVD) model and tested it to predict the rest of the unknown ratings. As shown in Figure 2.2, in the input layer, we passed the overall ratings that were conducted by the user for an item, then we used a normalisation equation to put those rating values between 0 and 1 (Eq.2.2). Followed by a number of hidden layers, we used the Rectified Linear Units (ReLU) as an activation function (Eq.2.1). The output of this single criteria model is the predicted overall ratings.



Users	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	5	4	4	2	?
User 2	1	4	2	3	1
User 3	3	1	3	5	2
User 4	5	5	4	2	5
...	...	...	...	...	...

Target User
Similar User
Ratings to be predicted
Ratings will be used in prediction

Figure 2.2: SVD Model Example..

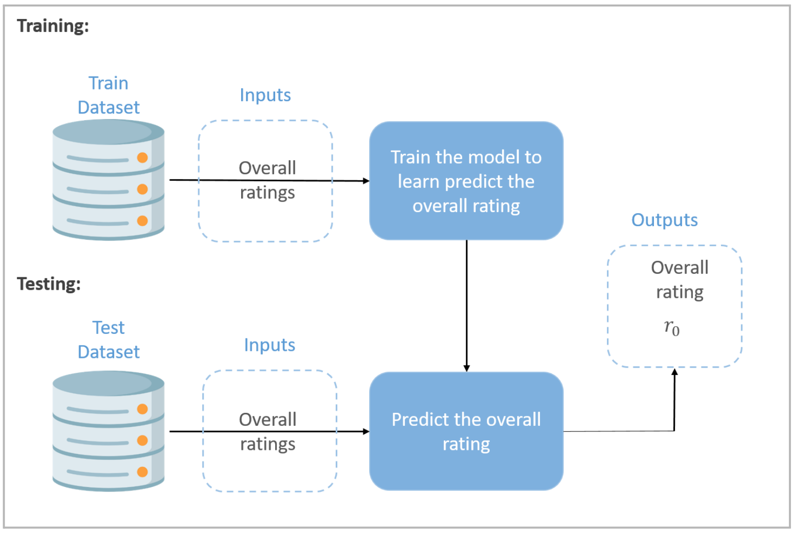
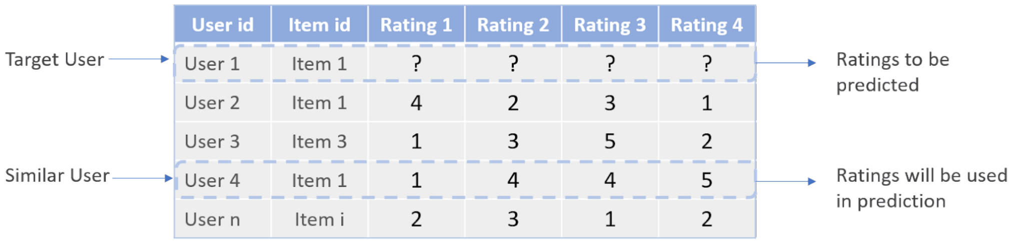


Figure 2.3: SVD approach.

$$\mathrm{ReLU}(v) = \mathrm{Max}(v,0) \quad (2.1)$$

Next, we implemented an MSVD model to predict the criteria ratings. The MSVD is used in multi dimensional datasets, where it learns to detect relationships between users and items, and uses them to predict criteria ratings to a target user as shown in Figure 2.4.



User id	Item id	Rating 1	Rating 2	Rating 3	Rating 4
User 1	Item 1	?	?	?	?
User 2	Item 1	4	2	3	1
User 3	Item 3	1	3	5	2
User 4	Item 1	1	4	4	5
User n	Item i	2	3	1	2

Target User
Similar User
Ratings to be predicted
Ratings will be used in prediction

Figure 2.4: Prediction in multi criteria dataset.

Model 2: Multi criteria using MSVD approach.

To predict the missing criteria ratings in this model we used the users and items ids as inputs to the model, As shown in Figure 2.5. Then we conducted a number of hidden layers, where each output of a hidden layer is the input to the next layer. We used ReLU as an activation function (Eq.2.1) because it gave the most efficient results. The output of this model is the predicted criteria ratings.

$$x' \; normalised = \frac{(x - x \; minimum)}{(x \; maximum - x \; minimum)} \quad (2.2)$$

Where the minimum and maximum values in the dataset are denoted by x minimum and x maximum, and x is the variable that holds the value (rating) we want to normalise. x’ is the normalised value.

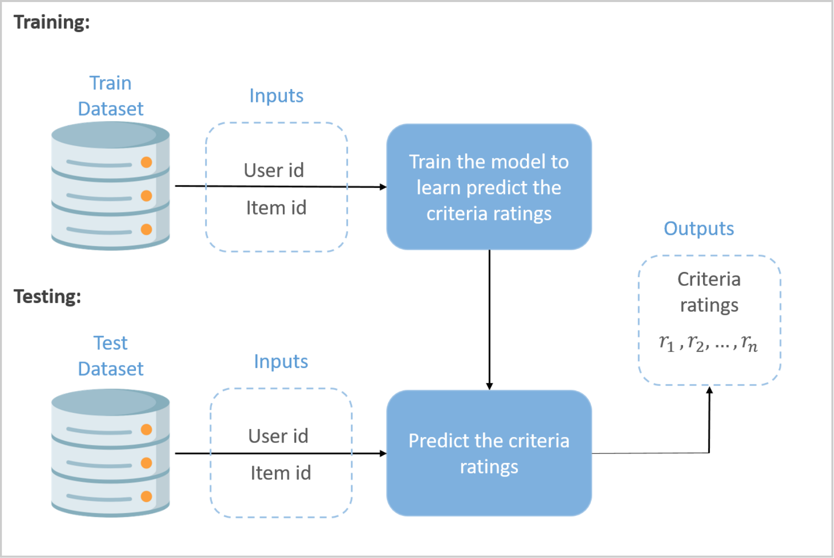


Figure 2.5: Representation of predicting criteria ratings with MSVD model.

Second, using the aggregation function we can predict the overall rating, which can be presented as follow:

$$r_0 = f(r_1, r_2, \ldots, r_n) \quad (2.3)$$

$$f_{avg}(r_1, r_2, \ldots, r_n) = \frac{1}{c}\sum_{i=0}^{c} r_i \quad (2.4)$$

Where r_0 is the overall rating, c refers to the number of criteria in each dataset and f is the aggregation function. We used the Average function as defined in (Eq.2.4) above.

Model 3: Multi criteria using Aggregation approach.

In the input layer, as shown in Figure 2.6, the inputs are the criteria ratings $r_1, r_2, \ldots, r_n$ for users on items, we normalised the values of those criteria between 0 and 1 using (Eq.2.2) to make it easy to train. We followed this process by a number of hidden layers, each output of a hidden layer is the input to the next layer, with ReLU as an activation function (Eq.1). Then, we used the average function defined in (Eq.2.4) to obtain the output of this model which is the overall ratings.

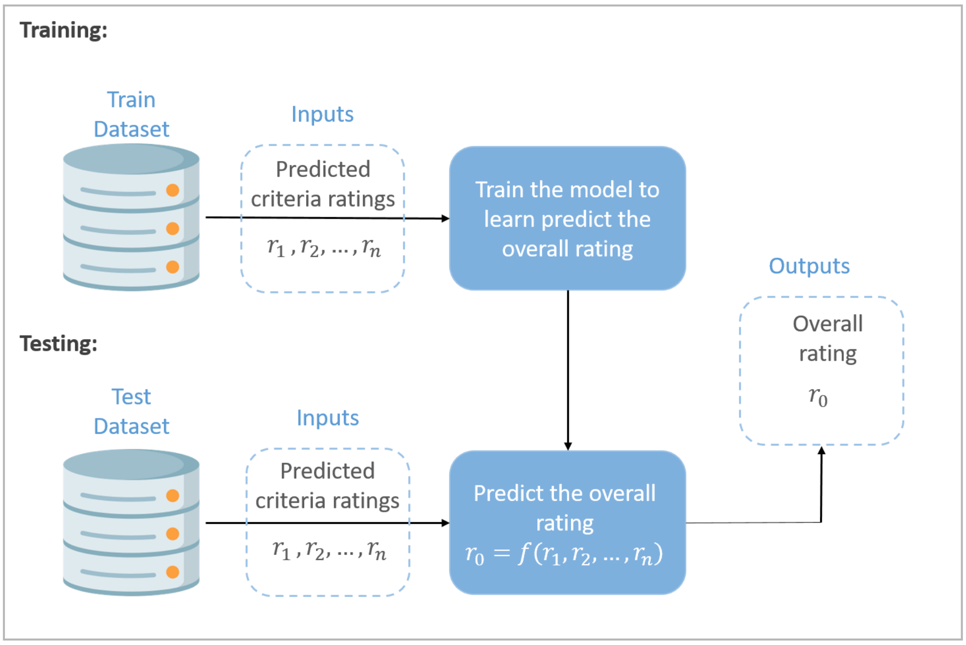

Training:
Train Dataset
Inputs
Predicted criteria ratings
$r_1, r_2, \ldots, r_n$
Train the model to learn predict the overall rating
Outputs
Overall rating
r_0
Testing:
Test Dataset
Inputs
Predicted criteria ratings
$r_1, r_2, \ldots, r_n$
Predict the overall rating
$r_0 = f(r_1, r_2, \ldots, r_n)$

Figure 2.6: Representation of predicting overall ratings with aggregation function.

1.2.2. Improving Basic Models Using Self-Attention:

To apply the Self-Attention in the previous models, we implemented a Self-Attention layer for every criteria input layer as shown in Figure 2.7, to add attention score to the ratings based on similarities.

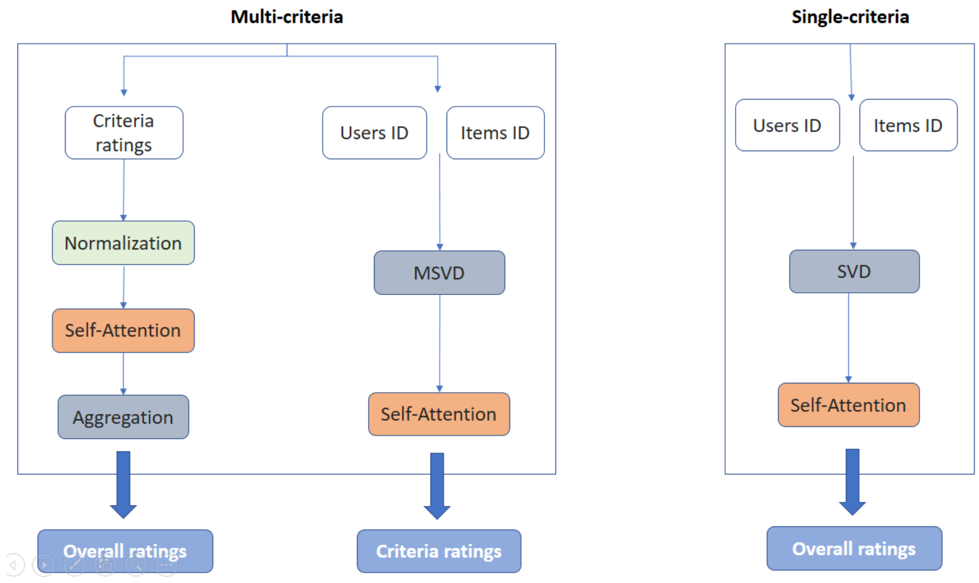


Figure 2.7: General architecture of our models.

Figure 2.7 above presents the general architecture of our models with Self-Attention.

And Figure 2.8 below represents the architecture of the Self-Attention layer that we used in our models.

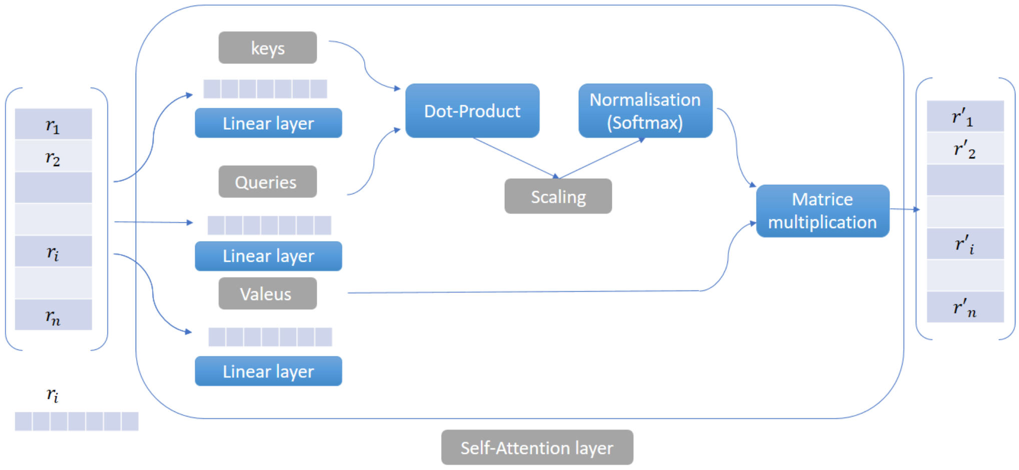


Figure 2.8: The architecture of the Self-Attention layer.

Model 4: Single criteria with Self-Attention using SVD approach.

This model is basically adding Self-Attention to Model1. In order to obtain the overall rating, we passed the users and items ids as inputs to the Singular Value Decomposition (SVD) model. Then pass them to a number of hidden layers, using ReLU as an activation function (Eq.1). Next we applied a Self-Attention layer to the outputs of the hidden layers . The output of this model is the predicted overall ratings.

Model 5: Multi criteria with Self-Attention using MSVD approach.

This model is adding Self-Attention to the second Model to predict multi criteria ratings using the MSVD approach. The users and items ids are the inputs of this model. We conducted a number of hidden layers, using ReLU as an activation function (Eq.1). Then, we applied a Self-Attention layer to those outputs. The output of this model is the predicted criteria ratings.

Model 6: Multi criteria with Self-Attention using Aggregation approach.

In this model we are going to add Self-Attention to the third Model to predict the overall ratings using the Aggregation approach. In the input layer we passed the criteria ratings of the users on items, we normalised the values of those criteria between 0 and 1 using (Eq.2). Then We represented each criteria as a vector and applied a Self-Attention layer to each criteria, as shown in Figure 2.9. We followed this by a number of hidden layers with ReLU as an activation function (Eq.1). Then, we used the average function defined in (Eq.4) to obtain the overall ratings.

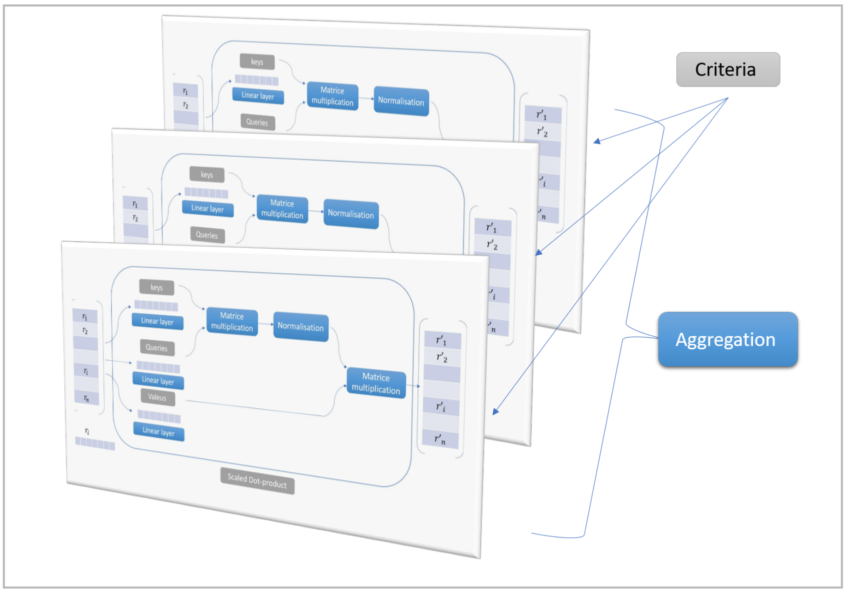


Figure 2.9: Aggregation approach with Self-Attention on each criteria.

1.2.3. Updating Previous Models Using Multi-Head Attention:

The multi-head attention, as described in the paper "Attention is all you Need" [2] and Afouras et al. [27], receives a query (Q), a key (K) and a value (V) tensor as inputs and then perform the self-attention function in parallel h times, one for every attention head i. However, increasing the number of heads does not necessarily lead to improved results. As shown in Figure 2.10. The Multi Head Attention formula is constructed as follows:

$$head_i = Attention(Q, K, V) \quad (2.5)$$

$$MultiHead(Q, K, V) = Concat(head_i, ..., head_h) \quad (2.6)$$

The multi-head attention allows the model to attend to information altogether from different representation subspaces at different positions. To ensure more variety, we expand the distances between attention heads. The number of heads in this model depends on the number of criteria in each dataset.

The query (Q) and key (K) are then dot-producted and scaled. These are softmaxed to obtain attention scores. The obtained matrices are then concatenated together producing the final attention matrix. To implement the Multi Head Attention in the previous models, the query, key, value must be the same Q = K = V, to be considered as Self-Attention.

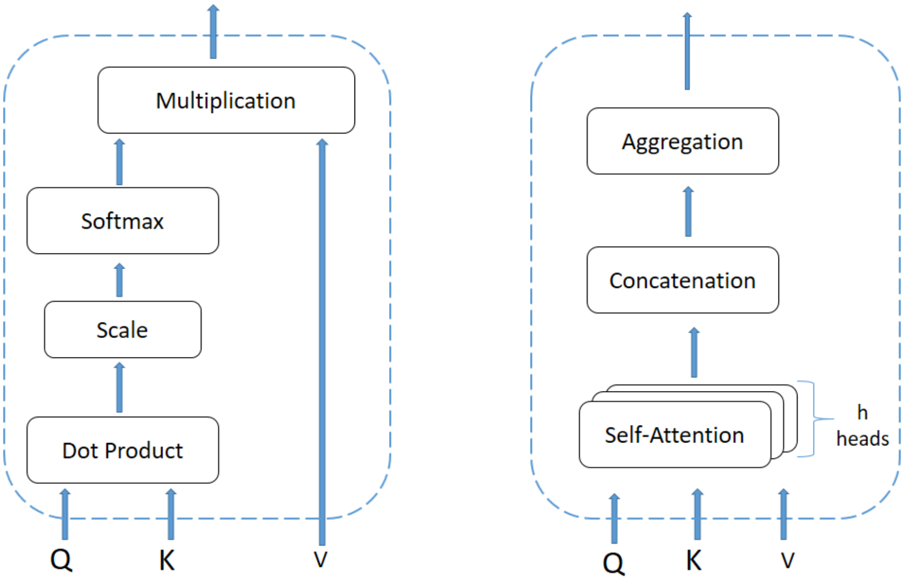


Figure 2.10: Multi Head Attention based on [27].

Model 7: Multi criteria with Multi Head Attention using Aggregation approach.

In this model we passed the criteria ratings as inputs to the multi head attention model, we used Eq.2 to normalise the data, we contacted those criteria to one vector to pass it to the Multi Head Attention layer. We initialised the number of attention heads according to the number of criteria of the dataset, the key dimension Size of each attention head for query and key. This layer takes 3 inputs: query, key and value, the key is Optional, if it's not given, the value will be used for both key and value.

2. Conclusion:

In this chapter we presented the proposed models of our study, starting from basic models with single and multi-criteria, then improved those models using Self-Attention. Next, we updated the previous models using Multi Head Attention. In the next chapter, we will present the results of those models to see the impact of the Self-Attention mechanism on multi-criteria recommendation.

Chapter 03

Experiments & Results Discussions

In this chapter, we will discuss the experiments and the obtained results of our work. We start with a brief description of the datasets that we used for testing our models. Then, we present the evaluation metrics used for evaluating the efficiency and quality of the models in addition to the implementation details. Next, we will discuss the experiments and present the obtained results by answering the questions that we faced during the implementation of the datasets in the models that we proposed in the previous chapter. At last, we conclude this chapter by a conclusion.

1. Datasets:

In this experiment, a large number of samples are used for training the models, we used two sparse databases that support multi-criteria recommendation. A hotel and a movie recommendation dataset called TripAdvisor and MoviesData, where users rate hotels and movies using a multi-criteria rating technique.

These public databases are provided online for free access, however, the TripAdvisor dataset needed preprocessing, to remove all unnecessary comments and information that makes it larger and takes too much time when working with it. We splitted the data randomly into 90% of samples from each dataset for training the models, and the rest 10% is used for testing it.

1.1 Statistics of The Datasets:

To provide further statistical information on the data. We demonstrate the features of both TripAdvisor and Movies dataset in Table 3.1 below.

Dataset Name	**Nb Users**	**Nb Items**	**Nb Ratings**	**Nb Criteria**	**Rating range**	**Sparsity**
TripAdvisor	80119	2724	101530	7	1 - 5	99.95%
Movies Dataset	6087	976	62156	4	1 - 13	98.95%

Table 3.1: Statistics of TripAdvisor and Movies dataset.

The table above presents a simple statics of both datasets, where TripAdvisor[1] dataset has 80119 users and 2724 hotels, it contains 101530 users' ratings for the hotels based on seven criterias ratings which are: Value, Rooms, Location, Cleanliness, Check in/front desk, Service, and Business service in addition to an overall rating. The ratings in this dataset are confined between one and five, with one being the lowest rating and five is the highest. The movies dataset is available online in GitHub[2], it has 6087 users and 976 movies, it contains 62156 users' ratings for these movies based on four different criterias ratings which are Value1, Value2, Value3 and Value4 as well as an overall rating. The ratings in this dataset are between one and thirteen, where one is the lowest rating and thirteen is the highest. The value zero in both datasets indicates a missing or unknown rating. The data sparsity percentage in both datasets was calculated based on [16] using this formula:

[1] https://www.tripadvisor.com/

[2] https://github.com/an888ha/multi_criteria_recommender_system/blob/master/data_movies.txt

$$Sparsity\ (\%) = \frac{((Nu * Ni) - Nr) * 100}{Nu * Ni} \quad (3.1)$$

Where Nu refers to the number of users, Ni is the number of items and Nr is the number of ratings.

1.2 Criteria Sparsity Problem:

Through our preprocessing on the TripAdvisor dataset we found that there are some criterias that are rarely evaluated by the users which are check in/front desk and business service. And this may cause a reduction in the quality of the prediction. We can call this problem the criteria sparsity problem.

Criteria	Value	Rooms	Location	Cleanliness	Check in / front desk	Service	Business service
Sparsity	10.8%	18.1%	24.8%	10.4%	79.4%	10.2%	96.8%

Table 3.2: Sparsity level in each criteria in TripAdvisor dataset.

The table above presents the sparsity percentage in each criteria in the TripAdvisor dataset. The sparsity was calculated using the same formula that was used earlier. We can see that the seven criterias have a different level of sparsity where some criterias have a low level of sparsity which are Value by 10.8%, Rooms by 18.1%, Location by 24.8%, Cleanliness by 10.4% and Service by 10.2% as the lowest value. As for the other criterias they have a high level of sparsity which are Check in / front desk by 79.4% and Business service by 96.8% as the highest value. These statistics indicate that these last two criterias with the highest level of sparsity may cause a limitation in the recommendation system.

2. Evaluation Metrics:

Evaluation metrics measure how much the recommender system's predicted rating differs from the user's actual rating. There are two types of metrics: Statistical metrics that can be used in the prediction phase and Decision making metrics that are used in the recommendation phase where a list of recommendations is provided for the user to choose the most suitable item. To evaluate the performance accuracy of our models we adopted several metrics which allow us to measure the error percentage that the system can encounter during the prediction. With no standardised metrics in the recommendation field, new evaluation metrics continued to appear. Therefore in this study we used these three most used metrics we encountered during our search: [24, 25].

Mean Absolute Error (MAE): This metric measures the average of the absolute deviance between the system's predicted rating and the actual rating of the users. The MAE equation is given as follow:

$$MAE = \frac{\sum_{i=1}^{N} \left| pr_i - ar_i \right|}{N} \quad (3.2)$$

Mean Square Error (MSE): This metric squared the average of the deviance between the predicted rating and the actual rating, to give more importance to higher deviances. The MSE equation is given as follow:

$$MSE = \frac{\sum_{i=1}^{N} \left(pr_i - ar_i\right)^2}{N} \quad (3.3)$$

Root Means Square Error (RMSE): Another metric for measuring the prediction accuracy is the RMSE. where we take the MSE and apply the square root on it, to emphasise more on the error. The RMSE equation is given as follow:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} \left(pr_i - ar_i\right)^2}{N}} \quad (3.4)$$

Where pr_i refers to the predicted rating for a user on an item i, ar_i is the actual rating of the user on an item i and N is the total number of ratings.

3. Implementation Details:

For implementing our models, we used Python as a programming language in addition to libraries such as numpy, tensorflow, keras. We used Google Colaboratory as a browser based environment for writing and executing our code. We trained our models with Adam, Adamax and RMSprop optimizer for both datasets, using batch size of 64 and set epochs to 30. We implemented from 8 to 256 hidden layers. To improve the model's performance, we applied the Rectified Linear Units (ReLU) between the layers, which is one of the most common and efficient activation functions. Since our models contain two branches: predicting criteria ratings and predicting the overall rating, The number of neurons in the output layer of each model is the number of the criterias and the overall rating.

4. Experiments and Results:

In order to study the impact of Self-Attention in the prediction phase in a multi-criteria collaborative recommendation system, we have to answer those following questions:

1. Is the Self-Attention mechanism efficient in reducing data sparsity problems?

To evaluate how self attention affects data sparsity problems we created a model where we passed the datasets as inputs to the self attention model and then calculated the sparsity of the output results. In the table below we compared the percentage of sparsity in each dataset before and after applying self attention.

Datasets	Sparsity without Self-Attention	Sparsity with Self-Attention
TripAdvisor	99.95%	91.53%
Movies	98.95%	84.56%

Table 3.3: Comparison of datasets sparsity in the model with/without Self-Attention.

Table 3.3 contains a comparison between the data sparsity level in both TripAdvisor and Movies dataset with and without the Self-Attention. Where we can see that the sparsity in the TripAdvisor dataset was 99.95% but after applying the Self-Attention to it, it reduced by 8.42% to become 91.53%. And as for the Movies dataset the sparsity was 98.95% and after the Self-Attention it reduced remarkably by 14.39% to become 84.56%. This results shows that the Self-Attention has significantly reduced the data sparsity problem in both datasets. Therefore we can say that Self-Attention is efficient in reducing data sparsity. But is it also effective when it comes to prediction?

2. What is the impact of Self-Attention on the quality of the prediction?

To assess the impact of Self-Attention on the quality of the prediction, we started by a comparative table (Table 3.4) on a single criteria SVD model and multi-criteria Aggregation model, using the overall rating of the multi criteria datasets. The obtained results are as follows: (these results are tasted with [64, 32, 16, 8] hidden layers).

			Loss error in different optimizers		
Models	**Datasets**	**Matric**	**Adam**	**Adamax**	**RMSprop**
Single criteria	**TripAdvisor**	MAE	0.1288	0.1092	0.1119
		MSE	0.0221	0.0200	0.0210
		RMSE	0.1521	0.1406	0.1440
	Movies	MAE	0.1633	0.1912	0.1706
		MSE	0.0481	0.0510	0.0510
		RMSE	0.2176	0.2511	0.2248
Multi criteria	**TripAdvisor**	MAE	0.0729	0.0729	0.0639
		MSE	0.0051	0.0076	0.0054
		RMSE	0.0716	0.5181	0.0739
	Movies	MAE	0.0573	0.0716	0.0598
		MSE	0.0088	0.0103	0.0093
		RMSE	0.0941	0.0956	0.0961

Table 3.4: Experimental results for the model in single and multi criteria with different optimisers.

The results of the table above prove that using multi-criteria recommendation enhances the accuracy of the recommendation system in ratings prediction.

Next, we will apply Self-Attention on single criteria SVD model and multi-criteria MSVD model. The results of this experiment are shown in Table 3.5 below.

			Accuracy in different optimizers		
Models	**Datasets**	**Matric**	**Adam**	**Adamax**	**RMSprop**
Self-Attention in Single Criteria SVD	**TripAdvisor**	MAE	0.1578	0.1102	0.1098
		MSE	0.0344	0.0203	0.0201
		RMSE	0.1925	0.1413	0.1411
	Movies	MAE	0.2033	0.2008	0.1984
		MSE	0.0730	0.0702	0.0667
		RMSE	0.2692	0.2633	0.2568
Self-Attention in Multi Criteria Aggregation	**TripAdvisor**	MAE	0.0620	0.0620	0.0640
		MSE	0.0051	0.2685	0.0054
		RMSE	0.0722	0.0717	0.0736
	Movies	MAE	0.0548	0.0544	0.0599
		MSE	0.0091	0.0091	0.0095
		RMSE	0.7682	0.0920	0.7684
Self-Attention in Multi Criteria MSVD	**TripAdvisor**	MAE	0.1189	0.2252	0.2748
		MSE	0.2716	0.3237	0.3230
		RMSE	0.2713	0.2711	0.1150
	Movies	MAE	0.3880	0.2987	0.30143
		MSE	0.2049	0.2060	0.2956
		RMSE	0.2961	0.2980	0.3875

Table 3.5: Experimental results for the Self-Attention model in single and multi criteria with different optimisers.

As Table 3.5 shows we applied different metrics in different optimisers with a 0.001 learning rate, to test the accuracy of our models in both single and multi criteria. The results of this table shows the loss error of the model with three different optimizers and three metrics. We found that Self-Attention reduced the loss error of the model, and the accuracy of the model is enhanced when using Adam optimiser, therefore we will select this optimiser for the rest of the experiments part. But what is the effect of the number of the hidden layers on the accuracy of this model?

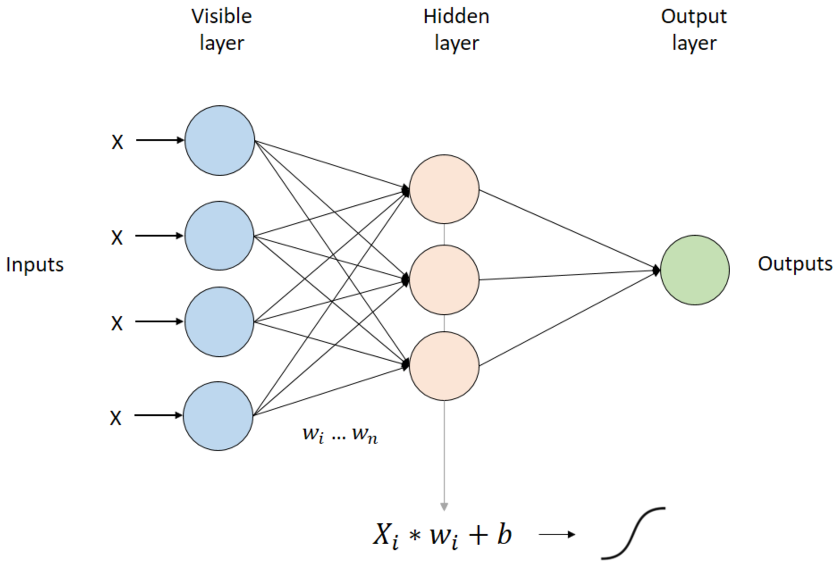


Figure 3.1: A Simple Artificial Neural Network (ANN) based on [18]

It consists of visible and hidden layers. The visible layer is the input layer that takes the raw data and passes them to the next layer. The hidden layer takes those input values and computes its output by applying weights calculation and bias. And then pass them through the activation function. This process is performed as follows:

$$Output = f(\sum_{i=1}^{n} x_i * w_i + b) \quad (3.5)$$

where x_i refers to the input value, w_i is the weights, n is the number of inputs on each neuron, b is the bias and $f()$ is the activation function.

To further improve our model, we tried different hidden layer configurations in the model to learn, as demonstrated in Table 3.6 below.

		Loss error in different hidden layers		
Datasets	**Matric**	**[64, 32,16, 8]**	**[128, 64, 32,16, 8]**	**[256,128, 64, 32,16, 8]**
Trip Advisor	MAE	0.0626	0.0640	0.0619
	MSE	0.0051	0.0051	0.0051
	RMSE	0.0715	0.0716	0.0694
Movies	MAE	0.0551	0.0563	0.0570
	MSE	0.0088	0.0091	0.0092
	RMSE	0.0937	0.0946	0.0949

Table 3.6: Experimental results for the Self-Attention model in different hidden layers configuration.

The table above shows the accuracy of the multi-criteria model in different hidden layers configuration in both datasets with three different metrics. As a result of this experiment, we selected the MSE evaluation metric and the [64, 32, 16, 8] hidden layers, because it gave us better accuracy than the other configurations.

In order to improve our model more, we need to study the effect of the criteria sparsity problem that raises more questions on how we can preprocess the data, and its impact on the accuracy of the model.

3. Do the sparse criterias affect the quality of the prediction? and what are the possible suggested solutions to reduce criterias sparsity?

Comparison table of using Self-Attention in normal TripAdvisor dataset and sparse-criteria TripAdvisor dataset.

			Loss error in Adam optimizer	
Models	**dataset**	**Matric**	**Aggregation**	**MSVD**
Self-Attention in normal dataset multi-criteria	**TripAdvisor**	MSE	0.0051	0.2716
Self-Attention in 1M dataset multi criteria	**TripAdvisor 1.6M**	MSE	0.0147	0.1753

Table 3.7: Loss error of Self-Attention in Aggregation and MSVD model.

In order to study the effect of those criterias on the performance of the model we suggest this experiments for TripAdvisor dataset:

- **The First Experiment: Remove the sparse criterias:**

In this experiment we removed the columns of the sparse criterias from the input dataset (TripAdvisor 1M). This experiment was conducted to see if they affect the quality of the prediction. The table below shows the accuracy of the model after removing the sparse criteria.

Datasets	**Matric**	**Aggregation**	**MSVD**
TripAdvisor	MSE	0.0139	0.1862

Table 3.8: Loss error of the Self-Attention model.

This experiment presents the accuracy of the Self-Attention model in the TripAdvisor 1M dataset, we tested this model with MSE evaluation metric using Adam optimiser, and recorded its accuracy in Table 3.8.

- **The Second Experiment: Fill the sparse criteria with the average (median) of the rated ones:**

In this experiment we took the available ratings of the Check in / front desk and Business service criterias and calculated their median. The obtained result was added to the unrated cells. The purpose of this experiment is to enhance the model’s performance, taking into consideration the average known rating of the users as a rating for the rest of users that didn’t rate those two criterias.

Datasets	Matric	Aggregation	MSVD
TripAdvisor	MSE	0.0237	0.2916

Table 3.9: Loss error of the Self-Attention model.

This experiment presents the accuracy of the Self-Attention model in the TripAdvisor dataset, where we tested the model with MSE evaluation metric, using Adam optimiser. Table 3.9 shows the results of this experiment.

- **The Third Experiment: Fill the sparse criteria with the users overall rating :**

In this experiment we replaced the unknown ratings of the Check in / front desk and Business service criterias with the overall rating of each user. This experiment aims to enhance the model’s performance, by taking the overall rating of the users into consideration in the prediction phase.

Datasets	Matric	Aggregation	MSVD
TripAdvisor	MSE	0.0012	0.1615

Table 3.10: Loss error of the Self-Attention model.

The table above presents the accuracy of the Self-Attention model in the TripAdvisor dataset, with MSE evaluation metric, using Adam optimiser.

4. What is the impact of those suggestions on the prediction’s accuracy?

After conducting those experiments we can select a solution to the sparse criterias that are affecting the prediction accuracy. The impact of each experiment was different from the others. Where the first and second experiments didn’t perform well according to the third experiment. The previous tables of the proposed solutions' accuracy shows that the third experiment has better accuracy results therefore using the overall rating of the user instead of sparse criteria has enhanced the quality of the prediction. This solution contributes to the development of more powerful recommendation systems that can significantly reduce the sparsity problem and thus further enhance the accuracy and efficiency of prediction.

5. Does using Multi-Head Self-Attention instead of Self-Attention affect the execution time of the prediction? And does the size of the dataset affect the performance of those models?

- **The Fourth Experiment: Multi Head Attention:**

In this experiment we conducted another model where we used the Multi Head Attention to evaluate the accuracy of the model compared to the previous Self-Attention model. The results of the Multi Head Attention model are presented in the table below.

Datasets	Matric	Aggregation Accuracy
TripAdvisor	MSE	0.0006
Movies	MSE	0.0091
TripAdvisor 1M	MSE	0.0062

Table 3.11: Loss error results of the Multi Head Attention.

This experiment presents the accuracy of the Multi Head Attention model in TripAdvisor and Movies dataset. We tested this model with MSE evaluation metric, using Adam optimiser, with the number of heads equal to the number of criteria in each dataset, the accuracy results of this experiment is shown in Table 3.11.

				Execution Time	
Models	**Datasets**	**Matric**	**Aggregation**	**In size of 1000 x 1000**	**In size of 1m x 1m**
Self Attention	**TripAdvisor**	MSE	0.0051	80sec	20min
	Movies	MSE	0.0091	40sec	//////////////////////
Multi Head Attention	**TripAdvisor**	MSE	0.0006	118s	31min
	Movies	MSE	0.0091	76s	//////////////////////

Table 3.12: Loss error results and execution time for the Self-Attention and Multi-Head model with different sizes of data.

The table above presents a comparison of the execution time of two models in TripAdvisor and Movies datasets. The TripAdvisor dataset was tested in two sizes.

This table shows that using Multi Head Attention improved the accuracy of the prediction. However, it takes more time than the Self-Attention model. The execution time in both models increases when it comes to larger datasets. Therefore, we can say that the Multi Head Attention improved the quality of the prediction.

5. Conclusion:

In this chapter, we presented the experiments we made in order to reduce the sparsity problem in the prediction phase. We started by presenting the datasets we used in addition to the metrics we used to evaluate the models. Then, we talked about the criteria sparsity problem in the datasets. Next, we presented the experiments and results of our model.

Conclusion And Future Work

Conclusion:

In this study, we proposed models with Self-Attention in the collaborative filtering recommendation using multi-criteria. The experiments were made in order to reduce the sparsity problem in the prediction phase. The results showed that using multi-criteria ratings outperformed the single criteria (overall rating), because the overall rating does not provide detailed information to help in understanding why the user prefers an item it just gives us an average rating on how much the user likes the item, unlike using multi-criteria ratings that enable us to accurately assess the similarity between two users /items. During the experiments we faced the sparse criteria problem, which led to performing more experiments to reduce the criteria sparsity. The best performance of the models was obtained when using Adam optimiser, with MSE evaluation metric in [64, 32, 16, 8] hidden layers, and replacing the sparse criteria with the user's overall rating. As a result of this study, we found that the data sparsity problem in the datasets was reduced due to the Self-Attention mechanism, where the sparsity level was 99% in TripAdvisor and 98% in movies dataset, and after applying the Self-Attention it became 91.53% in TripAdvisor and 84.56% in movies dataset. Therefore we can say that self attention is efficient in reducing data sparsity, and it has enhanced the accuracy and efficiency of prediction. These results show the advantage of using multi-criteria with Self-Attention in collaborative filtering recommendation systems.

As for the future works, we can add more experiments on the criteria sparsity where we can predict the sparse criteria ratings. Moreover, we can conduct more study on the impact of the Self-Attention and Multi Head in the aspect of the execution time and the dataset size.

References:

[1] Adomavicius, Gediminas, and Alexander Tuzhilin. "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 (2005): 734-749.

[2] Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems* 30 (2017).

[3] Kang, Wang-Cheng, and Julian McAuley. "Self-attentive sequential recommendation." *2018 IEEE international conference on data mining (ICDM)*. IEEE, 2018.

[4] Tao, Ye, et al. "TRec: Sequential recommender based on latent item trend information." *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020.

[5] Al-Ghuribi, Sumaia Mohammed, and Shahrul Azman Mohd Noah. "A Comprehensive Overview of Recommender System and Sentiment Analysis." *arXiv preprint arXiv:2109.08794* (2021).

[6] Adomavicius, Gediminas, Nikos Manouselis, and YoungOk Kwon. "Multi-criteria recommender systems." *Recommender systems handbook*. Springer, Boston, MA, 2011. 769-803.

[7] Isinkaye, Folasade Olubusola, Yetunde O. Folajimi, and Bolande Adefowoke Ojokoh. "Recommendation systems: Principles, methods and evaluation." *Egyptian informatics journal* 16.3 (2015): 261-273.

[8] Ricci, Francesco, Lior Rokach, and Bracha Shapira. "Introduction to recommender systems handbook." *Recommender systems handbook*. Springer, Boston, MA, 2011. 1-35.

[9] Chien, Yung-Hsin, and Edward I. George. "A bayesian model for collaborative filtering." *AISTATS*. 1999.

[10] Adomavicius, Gediminas, and Alexander Tuzhilin. "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 (2005): 734-749.

[11] Deshpande, Mukund, and George Karypis. "Item-based top-n recommendation algorithms." *ACM Transactions on Information Systems (TOIS)* 22.1 (2004): 143-177.

[12] Linden, Greg, Brent Smith, and Jeremy York. "Amazon. com recommendations: Item-to-item collaborative filtering." *IEEE Internet computing* 7.1 (2003): 76-80.

[13] Zhang, Shuai, et al. "Next item recommendation with self-attention." *arXiv preprint arXiv:1808.06414* (2018).

[14] Peter Bloem “Transformers From Scratch”. http://peterbloem.nl/blog/transformers . Amsterdam, Netherlands, 18 Aug 2019.

[15] Lv, Yanxia, et al. "AICF: Attention-based item collaborative filtering." *Advanced Engineering Informatics* 44 (2020): 101090.

[16] Nassar, Nour, Assef Jafar, and Yasser Rahhal. "A novel deep multi-criteria collaborative filtering model for recommendation system." *Knowledge-Based Systems* 187 (2020): 104811.

[17] Moqa, Salem, et al. "Multi-criteria Dual-View Attention Network for Rating Prediction." *2021 International Conference on Innovative Computing (ICIC)*. IEEE, 2021.

[18] Khan, Zahid Younas, et al. "Deep learning techniques for rating prediction: a survey of the state-of-the-art." *Artificial Intelligence Review* 54.1 (2021): 95-135.

[19] Sarwar, Badrul, et al. "Item-based collaborative filtering recommendation algorithms." *Proceedings of the 10th international conference on World Wide Web*. 2001.

[20] Wan, Shanshan, and Zhendong Niu. "A hybrid e-learning recommendation approach based on learners' influence propagation." *IEEE Transactions on Knowledge and Data Engineering* 32.5 (2019): 827-840.

[21] Batmaz, Zeynep, et al. "A review on deep learning for recommender systems: challenges and remedies." *Artificial Intelligence Review* 52.1 (2019): 1-37.

[22] Aggarwal, Charu C. "An introduction to recommender systems." *Recommender systems*. Springer, Cham, 2016. 1-28.

[23] Su, Khoshgoftaar. "Su X., Khoshgoftaar TM." *A survey of collaborative filtering techniques, Advances in Artificial Intelligence* 10.2009 (2009): 421425.

[24] Lerato, Masupha, et al. "A survey of recommender system feedback techniques, comparison and evaluation metrics." *2015 International Conference on Computing, Communication and Security (ICCCS)*. IEEE, 2015.

[25] Herlocker, Jonathan L., et al. "Evaluating collaborative filtering recommender systems." *ACM Transactions on Information Systems (TOIS)* 22.1 (2004): 5-53.

[26] Li, Jiacheng, Yujie Wang, and Julian McAuley. "Time interval aware self-attention for sequential recommendation." *Proceedings of the 13th international conference on web search and data mining*. 2020.

[27] Afouras, Triantafyllos, et al. "Deep audio-visual speech recognition." *IEEE transactions on pattern analysis and machine intelligence* (2018).

[28] Anusha Lihala. "Attention and its Different Forms". https://towardsdatascience.com/attention-and-its-different-forms-7fc3674d14dc .2019.

[29] Mohammadi Farsani, R., and E. Pazouki. "A transformer self-attention model for time series forecasting." *Journal of Electrical and Computer Engineering Innovations (JECEI)* 9.1 (2021): 1-10.