Integration of a recommender system into an online video streaming platform

Olfa Messaoud

Ministry of Higher Education and Scientific Research

-----*-----

University of La Manouba

-----*-----

National School of Computer Sciences

MEMORY

presented for the purpose of obtaining

Research Master Degree in Computer Science

Path: Smart Systems Branch: ILSI

Topic : Integration of a recommender system into an online video streaming platform

---

Elaborated By :

Olfa MESSAOUD

Host Organism :

LORIA & RIADI

Defended on in front of the honorable jury members :

President : Dr. Anja Habacha

Reporter : Dr. Ferihane Kboubi

Co-supervisor : Dr. Azim ROUSSANALY

Supervisor : Dr. Sonia BEN TICHA

Academic Year: 2019-2020

Integration of a recommender system into an online video streaming platform

Abstract

The ultimate goal of this project is to develop a recommender system for the SmartVideo platform. The platform streams different content of local channels for the Grand Est Region of France to a large public. So, we aim to propose a solution to alleviate the data representation and data collection issue of recommender systems by adopting and adjusting the xAPI standard to fit our case of study and to be able to represent our usage data in a formal and consistent format. Then, we will propose and implement a bunch of recommendation algorithms that we are going to test in order to evaluate our developed recommender system.

Keywords : Recommender Systems, Learning Record System (LRS), xAPI standard, data representation, data collection.

Résumé

Le but ultime de ce projet est de développer un système de recommandation dédié à la plateforme SmartVideo de diffusion de vidéo en ligne. En effet, la plateforme met à disposition diverses contenus des chaînes locales de la région Grand Est du France. Alors, nous allons présenter une solution pour alléger le problème de représentation et de collecte de données d'usages par adopter et ajuster le standard xAPI pour représenter et collecter les données de façon simple et formelle. Ensuite, nous allons proposer et implanter des algorithmes de recommandation que nous allons les tester pour évaluer notre système de recommandation.

Mots clés : Systèmes de recommandations, entrepôt de données (LRS), standard xAPI, représentation et collecte des données.

Supervisors' signatures

Dr. Azim Roussanaly

Dr. Sonia Ben Ticha

DEDICATION

To my dear parents, who sacrificed everything to make me thrive,
And who have always supported, assisted and encouraged me in every step,
To my brother and sister, for their beautiful words, love and support,
Words cannot express the gratitude, pride and the deep love i feel toward them,
To my friends, for their frequent support and help during all my studying years,
To my teachers, who have been so kind, comprehensive and gentle for all of us
To all those i love and respect,
I dedicate this work.

Olfa

ACKNOWLEDGEMENTS

This project would never be successful and honorable without the effective care and guidance of my supervisors who have shared with me their expertise and knowledge. I have to express my honest gratitude to their confidence and to the opportunity they offer me.

Therefore, my sincere appreciations go to Dr. Azim ROUSSANALY my supervisor from LORIA, for his trust, availability, guidance, assistance, support, favor and good humor. And, especially for his big contribution in order to make this work interesting and well qualified.

I would like to thank my pedagogic advisor, Dr Sonia BEN TICHA for the honour of supervising me and giving me all the support, encouragement, and availability beside her constructive criticisms that redound the wealth and the extension of this work to deliver my best.

I am extremely thankful also to my friends for their positive feedbacks and advises that I considered a lot to improve this project and to clarify ambiguous points. Offering a big amount of help and interest to the work, I moved on quickly to develop a good application.

At this stage, I ought to thank the honorable jury members for accepting to examine and evaluate my modest work. And, to all my professors for their infinite help and continuous support during these years that lead me to improve my skills and deliver a good product.

Contents

List of Figures

List of Tables

General Introduction

Recently, we got submerged by various choices while attempting to choose our interesting ones. For example, Which news or articles to read ? Which music to listen to or to see ? Which product to purchase ? Which movie to watch ? etc. The size of these decision domains is huge and massive in terms of used technologies, tools, and especially the used volume of data. And, helping the user to discover and pick out the pertinent resources is a challenging process.

As a result, the use of recommender systems became inevitable in such fields like online shopping websites, videos streaming platforms, etc. The ultimate success of some streaming platforms amounts to the quality of the used recommender system. For example, more than 80% of the watched streams on Netflix are discovered through the recommender [URL1]. And, 35% of Amazon Prime's revenue is generated by its recommendation engine [URL2].

We introduce SmartVideo project as a part of the CEI[1] project [URL3] consisting of developing a video streaming platform for the Grand Est Region of France. The platform is in the process of development by Kardham Digital [URL4] with the help of Via Vosges TV producers [URL5]. And, the KIWI team [URL6] of LORIA [URL7] participates to propose and develop the recommendation engine that will be integrated in the platform beside suggesting a solution to collect the usage data from the platform in real time.

Hence, we will first dive deeper into the Data Collection process of recommender systems in order to figure out the faced issues and to propose the best solution that will solve these issues. Then, we will present our proposed solution consisting of adopting the xAPI standard (see section 3.2). So, we will define the different applications, terminologies, benefits, and specifications of xAPI standard ending by proposing our specified xAPI video Recipe.

[1] Call for Expressions of Interest

Once is done, a thorough study will be done about recommender systems to present the non personalized and personalized recommendation algorithms, the faced challenges, the applications of recommender systems in Smart Video field, and the evaluation of these recommenders. Then, we will define a bunch of recommendation algorithms that we are going to implement, execute, and evaluate in order to pick out the best approach for our usage data.

The main work will be resumed as follows:

Chapter 1: Recommender Systems

The chapter will define the Recommender Systems presenting the most used non personalized and personalized recommendation algorithms. Then, it will detail the problems faced by recommender systems and how can they be managed. After, it will present the data collection process ending by defining the evaluation task of recommender systems.

Chapter 2: Smart Video

This chapter will focus on the Smart Video field by presenting the video streaming process. Also, it will introduce the most important video streaming platforms emphasizing how these platforms deal with data to make recommendations. Ending by presenting the recommendation process used for Smart Video field detailing some researches and works done.

Chapter 3: Data Collection

The chapter will detail the Smart Video Grand Est project. Then, it will present the adopted solution for dealing with the Data Collection process which consists of using the xAPI standard to present and manage the usage data. After, we will propose the xAPI video recipe ending by presenting the possible communication methods between xAPI and SmartVideo.

Chapter 4: Recommendation algorithms

This chapter will detail the non personalized as well as the personalized recommendation algorithms proposed and implemented for this project. Then, it will present the evaluation results applying our data. Besides, a discussion will be reported to interpret results, decide about the quality of recommendations, and figure out the best recommendation algorithms.

Chapter 1

Recommender Systems

A Recommender System (RS) is an information filtering system [Hanani et al., 2001] giving every user the most relevant information basing on his taste and preferences. The recommendation process can be split into non personalized recommendation where the recommended items are common for all or a group of users and personalized recommendation where the recommended items differs for every user.

The chapter will first define the non personalized recommender systems in the section 1.1, then will introduce the personalized recommender systems in the section 1.2, following by mentioning the problems of recommender systems in the section 1.3. After, it will present an overview about the data collection process for recommender systems in the section 1.4 ending by defining the evaluation process of recommender systems in the section 1.5.

1.1 Non personalized Recommender Systems

Non personalized recommender systems [Schafer et al., 1999] suggests identical recommendations to a group of users without taking into account the personal preferences and interests of each user. Although, these non personalized recommendations help creating a first impression about the users' prior interests and preferences to build the desired user profile that will be used in personalized recommendation.

We can find various recommendation techniques for non personalized recommendations depending on the user's usage and the stored information. Non recommender systems does

not require much information about the user, simply some demographic information, social information, or the IP address of the user are sufficient to perform a recommendation algorithm [Cremonesi et al., 2010].

Non personalized recommender systems are the most simple type of recommender systems. As suggested by the name, this type of recommender systems does not consider personal tastes of users. The recommendations produced by these systems are identical for every customer. This kind of recommendations is presented generally using a Top N list of recommended items [Poriya et al., 2014].

Non Personalized recommender systems mainly use two types of algorithms: Aggregated opinion recommender and Basic product association recommender.

1.1.1 Aggregated Opinion Approach

There are various online websites which make use of non personalized recommender systems by displaying the average customer ratings. Some of the famous website guides utilize the non personalized Aggregated Opinion Approach approach to suggest items (articles, videos, songs, movies, courses, restaurants, hotels, etc.) to users [Jones and Norrander, 1996]. The recommender system displays the items with a score.

This score is an average (mean) of the ratings given to every item by the customers. It is basically a measure to filter items and get the Top N list of highly scored ones. These scores generally range from 0 to 5. The average score is then calculated applying the formula 1.1 [Poriya et al., 2014].

$$Score = MEAN(ratings * 10) \tag{1.1}$$

Other examples that make use of aggregated opinion recommenders include travel websites (Trip advisor) which give reviews and ratings of different places around the world [Delgado and Davidson, 2002] or movie websites which display movie charts with Top N movies (having the highest average ratings) [Masthoff, 2015]. Averages are useful for an overall sense of what the group of individuals feel.

Besides, [Castro et al., 2017] proposes a group Recommender System based on Opinion Dynamics [Hegselmann et al., 2002] that consider relationships between members' preferences, which improves the aggregation. And, [Ben-Arieh and Chen, 2006] introduced a new linguistic-labels aggregation operation to handle the autocratic group decision-making process under linguistic assessments.

However these averages lack context during recommendation. One bad rating or review has a lot of weight and can pull down an otherwise excellent rating. To tackle this issue, some websites tallied the percentage of people who rated a particular item as « good » or « bad ». This leads us to the concept of product association recommender.

1.1.2 Basic Product Association Recommender

Basic product association recommenders [Konstan, 2008] can provide useful non personalized recommendations in a context. Majority of the online shopping websites such as Amazon or Flipkart make use of product association recommenders by providing « people who bought item 1 also bought item 2 » feature. Such recommendations are based on what is present in the user's chart [Demiriz, 2004].

Recommendations may not be necessarily specific to the user but specific to what the user is currently doing (viewing, buying). The basic idea for these systems is: People who did some X also did Y. The simple computation of this ranking can be the percentage of X-buyers who also bought Y. This is illustrated in the formula 1.2.

$$Ranking = \frac{XandY}{X} \tag{1.2}$$

However, the formula 1.2 does not compensate for the overall popularity of Y so it is adjusted in the following manner by looking at whether X makes Y more likely than not X (!X) which is represented in the formula 1.3

$$Ranking = \frac{\frac{XandY}{X}}{\frac{!XandY}{!X}} \tag{1.3}$$

The formula 1.3 focuses on increasing the Y associated with the X [Poriya et al., 2014]. Another solution to the drawback of this formula is using the Association rule mining which uses the lift metric (Likelihoods) [Mukhopadhyay et al., 2008]. It is basically non-directional

association. More generally, association rules look at baskets of products, not just individuals [Anderson and Hiralall, 2009]. The formula 1.4 is:

$$Ranking = \frac{P(XandY)}{P(X) * P(Y)} \tag{1.4}$$

Also, many other non personalized recommendation algorithms can be applied such as Most Popular, Recently Added, Recently Viewed, Most Viewed, Most Rated, Best Rated, Most Commented, Most Shared, Most Clicked, Nearby Items, etc. These techniques are based on different parameters like geography (IP address), age, sex, release date, ratings, comments, shares, clicks, etc. [Khatwani and Chandak, 2016]

The advantage of the non personalized recommender approaches is that they are easy to implement as only the popular or the highly rated items are displayed to the users. Secondly the data for these recommender systems are widely available and easy to collect [Iyer et al., 2015]. In particular cases, a naive non personalized algorithm can outperform some common recommendation approaches.

Although, these systems face challenges in clustered diverse population such as the example of the banana trap problem [Poriya et al., 2014]. Also, Recommendations are the same to all users and lack personalization and hence might not appeal to everyone. That's why, the ultimate goal of using non personalized algorithms is to provide a baseline for more sophisticated personalized algorithms.

1.2 Personalized Recommender Systems

A Personalized Recommender [Zhou et al., 2012] system can be defined as a program that attempts to recommend the most suitable items to a particular user by predicting his interests basing on the historical information about his tastes and preferences. The asset of personalized recommender systems is that they focus on examining the data issued from the users' usage or interactions than exploiting data to perform the recommendations.

Personalized recommender systems uses three kinds of information [Jannach et al., 2010]:

- **Information about users:** The users of the system can be defined by their demographic

informations (age, gender, marital status, education, employment), interests and preferences, and other personal information.

- **Information about items:** The items are the elements that are recommended to the users by the recommender system. They can be news, journals, films, videos, restaurants, books, articles, etc. An item can be described by its identifiant, name, description, category, release date, duration, keywords, etc.
- **Information about preferences:** In order to make a personalized recommendation for every user, the preferences are used to describe the personal tastes and interests of every user. The preferences can be explicitly detailed using a rating, an annotation, a Like/Dislike, etc. [Brusilovsky and Millán, 2007]. Or, they can be implicitly presented by the comportment and the activities done by the users like the historical search and consultations, the number of clicks on an item, etc. [Anand and Mobasher, 2003].

The personalized recommender systems became very popular and widespread to be applied in various e-services such as e-commerce, e-learning, etc. Hence, the use of efficient and accurate recommendation techniques is very important to increase the performance results of recommender systems which reflects the understanding of the features and the potential of different recommendation techniques [Aggarwal et al., 2016].

The personalized recommendation process is represented by collecting data related to users, items and their interactions, then analyzing these data and transforming them, so the personalized recommender learns from every user interests and preferences and predicts the most pertinent items willing to interest him. In fact, a bunch of personalized recommendation approaches were defined in order to process recommendation [Lu et al., 2015].

Mainly, three approaches are frequently used for recommendation using the usage data [Adomavicius and Tuzhilin, 2005b].

- **Content Based Filtering:** Recommends similar items to the items appreciated by the current user [Peis et al., 2008].
- **Collaborative Filtering:** Bases on the appreciations of a set of users on items, to make either User Based recommendations [Resnick et al., 1994] or Item Based recommendations [Sarwar et al., 2001].

- **Hybrid Filtering:** Combines different approaches to alleviate the issues resulted from the Content Based Filtering approach and the Collaborative Filtering approach [Burke, 2007].

1.2.1 Personalization

A personalized recommender [Gretzel and Fesenmaier, 2006] exploits different types of data in order to make recommendations. Data can be relative to items, users, and usage formed from the interactions or activities of the users to the items while using the e-service. These data are used differently depending on the utilized recommendation approach
[Adomavicius and Tuzhilin, 2005a].

1.2.1.1 Items data typology

An item is generally represented by a profile or a description that details the characteristics, attributes and properties of the item. For example, for a movie recommender system, items are presented by their identifiants, titles, genres, production date, producers, actors, etc.

The items can be represented by a textual form that refers systematically to the collected material consisting of written, printed, or electronically published words, sentences and paragraphs [Ekstrand et al., 2011]. Or, they can be represented by a multimedia data types (semantic data) like text, audio, video [Lops et al., 2011].

Semantic data can be structured, semi structured or non structured.

- **Structured data:** are highly organized and easily defined by a common limited number of attributes (e.g. title, date, gender, etc.) [Pazzani and Billsus, 2007].
- **Semi structured data:** are data having a free text area description (e.g. synopsis, resume, newspaper electronic article, etc.) [De Gemmis et al., 2015].
- **Non structured data:** are data that combine both structured and semi structured types (e.g. title, production date, synopsis, resume, etc.) [Lops et al., 2011].

Non structured data requires a content analysis phase to understand the data and convert them to a structured consumable form. Every long text area will be presented by a limited list of keywords or concepts. Besides, the non textual multimedia data are converted to vector

patterns having different modalities (I_V, I_A, I_T) where I_V, I_A, I_T are respectively the visual, audio and textual modalities [Deldjoo et al., 2018].

1.2.1.2 Users data typology

The user profile consists on describing the available features about users like demographic data (age, gender, localisation), identification data (mail, name, username, identifiant) and interests (tastes, preferred genres, ..). A lot of approaches are present to acquire the user profile, we can note the manual, automatic and semi automatic approaches [Burke, 2002b].

A manual approach is presented by filling a form with the needed data (age, gender, study, job, address, mail, etc.). Every set of data serves for a recommendation technique (i.e. demographic data are used in demographic based recommendations [Krulwich, 1997]). The principle problems of the manual approach is that the user is reticent and prevent giving any real information about him.

1.2.1.3 Usage Data analysis

In order to make a personalized modelisation of the preferences, we need first to know the user's tastes to recommend him the most favorable items that fit his preferences. The usage analysis process in recommender systems aims essentially to analyze the different interactions of the user with the e-service to infer his appreciations and preferences. These interactions are collected from different sources.

An interaction contains information about the user, the context description, and the appreciation consisting of whether the user is interested or not interested in the item that he have consulted. The appreciation is presented by a note, this note can be either a rating (vote) or an annotation (tag or comment) attributed explicitly or implicitly.

The notes can be explicit where the user types his own opinion, or implicit deduced from the user's actions toward the selected item. The notes allow reflecting a positive or a negative point of view of the user to the corresponding item. In fact, the ratings are more used than the annotations. So, ratings can be represented using one of the following forms [Schafer et al., 2007]:

- **Scalar Rating:** Numerical value defined using a scale of discrete values for example between 1 and 5 where 1 indicates that the user is unsatisfied and 5 indicates that the user is very satisfied.
- **Ordinal Rating:** Ordinal vote scale is described by an ordered and fixed list of values « Excellent, very good, good, bad, very bad ». The user chooses between these values that will be encoded numerically.
- **Binary Rating:** The Binary vote defines performed action of like or dislike of the user to the selected item.
- **Unary Rating:** The unary rating shows only if the user have observed, purchased, or evaluated an item. The absence of vote does not reflect any information about the relation between the user and the item.

The rating values are the most frequently data used in recommendation, these ratings are usually represented by a rating matrix where every line corresponds to the user, every column corresponds to the item and the case corresponding to the given user and given item contains the rating value. If there is no vote in the matrix, so the corresponding user has not yet rated the given item, which is called a missing value.

The figure 1.1 shows a simple example of the rating matrix used for storing the ratings of every user to every movie.

	Movie1	Movie2	Movie3	Movie4	Movie5
U1		5	4	2	1
U2	1			5	3
U3	1	4	4	1	
U4			2		2
U5	3	1	1		

Figure 1.1: Rating Matrix example

[URL8]

So, the non empty cases contain the rating values associated by the users to the movies, while the empty cases represent the non rated items (movies), these missing values increases exponentially with the increasing of the number of users and items which leads to the sparsity

problem of recommender systems [Chen et al., 2011].

The annotations or tags are considered as another form of evaluation allowing the user to explicitly express his opinion. The annotation is a string where the user puts whatever he wants freely in order to reflect his point of view on the given item. For example, MovieLens provides the possibility of giving an open space to the users to add their proper keywords or to choose from the annotation list [Ames and Naaman, 2007].

1.2.2 Recommendation

In recommender systems, a user profile is formed from the previous interactions of the user with the e-service. This user profile contains essentially the tastes and the preferences of the user and is elaborated for the purpose of serving recommendations. Such approaches were defined to represent and consume the formed user profile depending on the nature of the provided data.

1.2.2.1 Collaborative Filtering

Collaborative filtering [Thorat et al., 2015] utilizes previous interactions between users and items to generate new recommendations. The concept of collaborative filtering is using interactions to filter items basing on the assumption that if a user likes item A and another user likes the same item A as well as another item B, the first user will likely be interested in the second item B.

The collaborative methods are based on the historical interactions between users and items to generate novel recommendations. These interactions are essentially represented as the votes or ratings and are stored in the user-item rating matrix. Dealing with these user-item interactions (ratings) is sufficient to detect the similar users and/or items and to make predictions [Lousame and Sánchez, 2009].

The collaborative algorithms are separated in two categories: Memory Based and Model Based Collaborative Filtering algorithms [Su and Khoshgoftaar, 2009].

- **Memory Based algorithms:** These algorithms are based on heuristics [Adomavicius and Tuzhilin, 2005b] or on neighbors [Desrosiers and Karypis, 2011]. They

function directly with the registered votes on memory (rating matrix) and they focus essentially on searching the nearest neighbors (e.g. searching the nearest users from those having the same taste as the current user and suggest the most popular items) to apply prediction [Aggarwal et al., 2016].

- **Model Based algorithms:** These methods utilize an offline reduced image of the rating matrix to optimize the computational time and to deal with the missing values. Many approaches were used such as dimension reduction methods based on the linear algebra and Singular Value Decomposition (SVD-ModelBased) [Baroni et al., 2010], probabilistic approaches [Frazier et al., 1998], clustering based approaches [Ungar and Foster, 1998], and association rules based methods [Brill et al., 2001].

Memory Based Collaborative Filtering

The collaborative filtering memory based algorithms used the complete rating matrix or a part of it to apply recommendations. Collaborative algorithms are divided in two categories: User Based algorithms and Item Based algorithms [Adomavicius and Tuzhilin, 2005a].

- **User Based algorithms:** User-based predicts the potential interest of the user to an item using the neighbors' votes which are the users that have similar preferences as the given user [Resnick et al., 1994, Billsus et al., 1998].

- **Item Based algorithms:** Item-based determines the interest of the user to the item candidate for recommendation by using the ratings of the given user to the neighbors of the corresponding item [Sarwar et al., 2001, Linden et al., 2003].

The Memory Based and Model Based recommendation algorithms of the Collaborative Filtering recommendation technique will more detailed precisely in the chapter 4 specifically in the section 4.5.

1.2.2.2 Content Based Filtering

Contrarily to collaborative filtering recommendation methods based on user-item interactions only, Content Based methods [Lops et al., 2011] recommend items having similar description to the items a user have appreciated in the past. So, a content analysis process is done to identify the items liked by the user then define the items having similar descriptions that are

willing to like the current user.

The content based recommender systems exploit essentially the semantic content of the items to perform recommendation contrarily to collaborative recommendation methods that focus on interactions between users and items [De Gemmis et al., 2015]. Thus, identifying common characteristics to the favorably evaluated items by the current user is required to form a model based for the available characteristics [Deldjoo et al., 2016].

Content Based systems make recommendations using items content and profile features. They reason that if a user was interested in an item in the past, he will keep being interested in it in the future. So, the technique returns the similar items to the corresponding interesting item basing on similar features. So, the content based recommender systems identify the common characteristics of the list of items appreciated by the given user [Pyo et al., 2013].

Many approaches were utilized to represent the user profile from the item content [Lops et al., 2011]. We can cite the Vector Space Model (VSM) where the user is represented by vector of weights defined in the same space representing the items. Every weight measures the importance of the corresponding term to the given user [Grčar et al., 2006].

[Musto et al., 2016] utilized the Word Embedding techniques [Ganguly et al., 2015] in a content based recommendation. [Zenebe and Norcio, 2009] applied a Fuzzy set Theoretic Method (FTM) that handles non-stochastic uncertainty induced from subjectivity, vagueness and imprecision in data. [Van den Oord et al., 2013] used a latent factor model [Jenatton et al., 2012] to predict non available music audios.

1.2.2.3 Hybrid Filtering

Content Based filtering requires the availability of semantic items following by a content analysis phase to extract the important data about items and present them. Unfortunately, content about items is insufficient, limited, and challenging to extract. Collaborative filtering systems does not take into charge the content which leads though to decrease the performance of the recommender [Burke, 2002a].

In order to overcome the problems of collaborative filtering and content based filtering techniques, the hybrid filtering approach is set to combine more than one filtering approach. The hybrid recommendation goal is to cope with some common issues of other recommendation techniques such as cold start problem, overspecialization problem and sparsity problem [Burke, 2007].

Also, hybrid filtering aims to improve the accuracy and efficiency of the recommendation process by combining multiple recommendation techniques together [Iaquinta et al., 2008]. A number of researches and works were done to ensure that different recommenders could work together (i.e. NewsDude, have used both naive Bayes and KNN classifiers in its news recommendations [Isinkaye et al., 2015]).

The possible hybridization methods are identified as follows [Thorat et al., 2015]:

- **Weighted:** The score of different recommendation components are combined numerically.
- **Switching:** The system chooses among recommendation components and applies the selected one.
- **Mixed:** Recommendations from different recommenders are presented together.
- **Feature Combination:** Features derived from different knowledge sources are combined together and given to a single recommendation algorithm.
- **Feature Augmentation:** One recommendation technique is used to compute a feature or set of features, which is then part of the input to the next technique.
- **Cascade:** Recommenders are given strict priority, with the lower priority ones breaking ties in the scoring of the higher ones.
- **Meta-level:** One recommendation technique is applied and produces some sort of model, which is then the input used by the next technique.

[Ghazanfar and Prugel-Bennett, 2010] proposed a cascading hybrid recommendation approach by combining the feature correlation with rating and demographic information of an

item. [Woerndl et al., 2007] applied a hybrid recommender system to deal with the added complexity of context integration in recommender systems. Users can select among several content based or collaborative filtering components.

[Melville-Jones, 2002] executed the collaborative filtering method on a defined user preferences matrix containing weighted keywords about users' tastes. [Pazzani and Billsus, 2007] utilized content based recommendations to complete the rating matrix then applied collaborative filtering recommendations on the resulted matrix.

1.3 Problems of Recommender Systems

This section investigates the issues and challenges in recommender systems [Khusro et al., 2016].

1.3.1 Cold Start Problem

Cold Start problem occurs when new users enter the system or new items are added to the catalogue. In such cases, neither the taste of the new users can be predicted nor can the new items be rated by the users leading to less accurate recommendations. Cold Start problem was tackled in several papers such as [Zhang et al., 2010], [Safoury and Salah, 2013], [Lika et al., 2014], [Pereira and Hruschka, 2015], and many others.

1.3.2 Synonymy

Synonymy arises when an item is represented with two or more different entries having similar meanings. In this case, the recommender system cannot identify whether the terms represent different items or the same item. For example, a content based recommender system will treat "comedy movie" and "comedy film" differently. The variation in using different synonym words decreases the performance of the recommender [Sarwar et al., 2000b].

1.3.3 Privacy

Feeding personal information about the users to the recommender system results in better recommendation services but may lead to privacy and security issues. That's why, users are reluctant to feed data into recommender systems [Khusro et al., 2016]. Some specialized

algorithms were proposed to deal with the privacy problem and build trust among users [Jeckmans et al., 2013, Aïmeur et al., 2008].

1.3.4 Overspecialization

Users are restricted only to get recommendations which resemble to those already known or defined in their profiles, it is termed as over specialization problem [Sharma and Gera, 2013]. It prevents user from discovering new items and other available options. However, diversity of recommendations is a desirable feature of all recommendation systems. The problem can be solved using genetic algorithms [Ar and Bostanci, 2016].

1.3.5 Limited Content Analysis

The limited availability of content leads to such problems because content is either scarce or challenging to obtain [Chidlovskii et al., 2001]. So, the limited content analysis leads to recommend closely related items to the user profile without taking into account the user preferences and do not suggest novel items [Hicken et al., 2005]. A content analysis phase should be done to deal with this problem [Davidson et al., 2010].

1.3.6 Grey Sheep

Grey sheep [Ghazanfar and Prugel-Bennett, 2011] occurs in pure Collaborative Filtering systems where opinions of a user do not match with any group so he can not get benefit of recommendations. Pure Content Based filtering can resolve this issue where items are suggested by exploiting user personal profile and content of items being recommended
[Ghazanfar and Prügel-Bennett, 2014].

1.3.7 Sparsity

The availability of huge size of data about items and the disinclination of users to rate items will give dispersed values leading to a big number of missing ratings values which gives less accurate recommendations. The sparse rating in CF systems makes it difficult to make accurate predictions about items. A lot of research papers tackled sparsity problem such as [Huang et al., 2004], [Papagelis et al., 2005], [Yildirim and Krishnamoorthy, 2008], etc.

1.3.8 Scalability

The rate of growth of nearest neighbour algorithms shows a linear relation with the number of items and the number of users. It becomes difficult for a typical recommender to process such large-scale data [Sarwar, 2001]. For example, Amazon recommends more that 35 million products to more than 150 million users in 2020 [URL9]. The scalability problem can be solved using an efficient recommendation algorithm [Bakshi et al., 2014].

1.4 Data collection for Recommender Systems

The first step in creating a recommendation engine is gathering data related to the interaction of the connected user with the system while performing a search or a recommendation task. Data can be structured or non structured, textual or non textual and is stored in several ways in different data sources. In recommender systems as in machine learning, data collection is a crucial task of Data Mining [Prakash and Rangdale, 2017].

In fact, most of the effort in data mining is always spent in cleansing and organizing data. Because garbage-in means garbage-out, we have to ensure that the data we are feeding to the recommender is clean and well structured resulting exact outputs. Data collection is integrated in the first phase of data mining workflow which is data processing containing the Extract, Load, and Transform stages (ETL) [Verma et al., 2015].

The typical ETL involves gathering data from multiple sources, data sanity checks, data transformation (structural changes, data standardization, enrichment, etc.) with the possibility of dropping and rejecting some source data depending on the business rules, and finally, loading the data to the target storage (generally data warehouses). The ETL traditional approach is presented in the figure 1.2

The Extract, Load, and Transform pipeline consists on [URL11]:

- **Extract data from different sources:** the basis for the success of subsequent ETL is to extract data correctly. Take data from a range of sources, such as APIs, relational or non relational databases, XML, JSON, CSV, log files, and convert them into a single format for standardized processing.

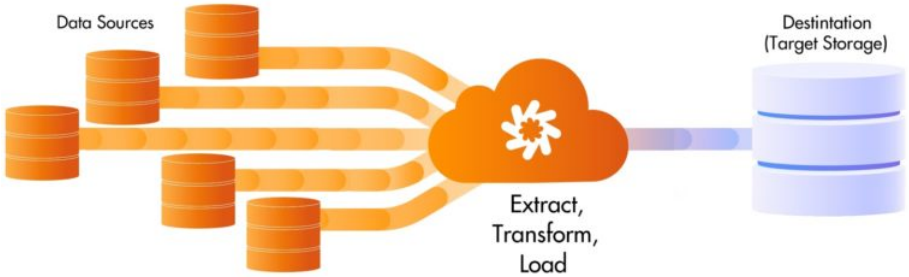


Figure 1.2: Extract, Transform and Load process
[URL10]

- **Validate data:** Keep data that have values in the expected ranges and reject any that do not. Analyze rejected records to identify issues, correct the source data, and modify the extraction process to resolve the problem in future batches.
- **Transform data:** Remove duplicate data (cleaning), apply business rules, check data integrity (ensure that data has not been corrupted or lost), and create aggregates as necessary. Numerous functions should have been implemented to transform the data automatically.
- **Stage data:** Data first enters a staging database which makes it easier to roll back if something goes wrong what leads to generate audit reports for regulatory compliance, or diagnose and repair data problems. Then, data is transformed directly into the target data warehouse.
- **Publish data in warehouses:** Load data to the target tables (daily, weekly, or monthly). The ETL workflow can add data without overwriting, including a timestamp, carefully to prevent the data warehouse from exploding due to disk space and performance limitations.

The first part of an ETL process involves extracting or collecting data from the source systems. Basically, this represents the most important aspect of ETL, since extracting data correctly sets the stage for the success of subsequent processes. A series of rules or functions are applied to extract data in order to transform them to a simple exploitable form and load them into the target sources.

The provided benefits of the ETL are [El Akkaoui et al., 2011]:

- **Ease of Use through Automated Processes:** The biggest advantage of ETL is its ease of use. After choosing the data sources, the tool automatically identifies the types and formats of the data, sets the rules how data has to be extracted and processed, and finally loads the data into the target storage which reduces coding.
- **Operational resilience:** ETL have a built-in error handling functionality which help data engineers to develop a resilient and well instrumented ETL process even when data warehouses are fragile during operation.
- **Good for complex data management situations:** ETL are great to move large volumes of data and transfer them in batches. In case of complicated rules and transformations, ETL simplify the task and assist with data analysis, string manipulation, data changes and integration of multiple data sets.
- **Advanced data profiling and cleansing:** refers to the transformation needs which are common to occur in a structurally complex data warehouse.

However, ETL process cannot keep up with the high speed of changes that is dominating nowadays especially with the upcoming of big data. Processing data with ETL means to develop a process with multiple steps so data need to be moved and transformed every time. Furthermore, ETL process is not only a costly process but also highly time consuming.

The major drawbacks of ETL process are [LANGab et al., 2008]:

- **Similarity of Source and Target Data Structures:** The more the source data structure differs from the target data, the more complex ETL processing and maintenance effort becomes.
- **Quality of Data:** Commonly, loaded data quality issues include missing values, code values, not correct list of values, dates and referential integrity issues.
- **Complexity of the Source Data:** Some data sources are complex including multiple record types, limited fields, multi-typed fields, missing values, not in range values, biased values, etc.

- **Cold start, warm start:** Unfortunately, systems do crash, the system could either starts over (cold start) or starts from the last known successfully loaded record (warm start).
- **Disk space:** Not only does the data warehouse have requirements for a lot of disk space, but there is also a lot of hidden disk space needed for staging areas and intermediate files.
- **Performance:** Extracting, transforming and loading the data needs a lot of computational time for the system that to collect data easily and correctly, then processing the cleaning phase that will take severe time.
- **Domain Knowledge:** The adoption of the ETL process needs a good expertise of the ETL domain how to extract, transform, and load data to the warehouse which is not a simple task.

1.4.1 Resort to xAPI

In order to avoid the whole time consuming ETL process while extracting data, transforming them, and loading them to datawarehouses, we opted for adapting the well known xAPI standard (presented and detailed in the chapter 3) basically introduced in the field of e-learning to represent the learning experiences of every user and his interactions with the e-service.

The xAPI standard [Manso-Vazquez et al., 2015] is the most used standard in the e-learning field to represent data in a simple structure. Hence, an adjustment of this xAPI standard is fulfilled to comply with our usage data while representing the different interactions of users to items. The upcoming of using xAPI is not only to avoid using the costly ETL process but also to reduce the time consuming Data Mining task [Yu et al., 2012].

Representing data uniformly using xAPI standard then storing them into data warehouses will evade such issues. So, we will be able to directly collect data from the data warehouse and preprocess data easily because we deal with well structured and organized data. Thus, the Extract, Transform, and Load tasks will be replaced simply by a data collection task.

1.5 Evaluation of Recommender Systems

Several studies have investigated how to measure the effectiveness of recommender systems in order to decide the best algorithm fitting the current case study. Recommender systems quality can be evaluated comparing recommendations to a notation (rating, tag) test set of known users. The evaluation of recommenders has two kinds: Online evaluation [Xiao et al., 2018] and Offline evaluation [Gilotte et al., 2018].

At the base of the vast majority of recommender systems lie a prediction engine. This engine may predict user opinions over items or the probability of usage. A basic assumption is that the recommender system providing more accurate predictions will be more preferred by the user. Thus, many researchers set out to find algorithms that provide better predictions.

In literature [Herlocker et al., 2004], many offline algorithms were proposed with the possibility of comparing the offline performance of such recommendation algorithms objectively. An interesting approach is to test different recommendation methods offline, then to evaluate online the best among them. A detailed study on the evaluation modes of recommender systems is cited in this article [Shani and Gunawardana, 2011].

1.5.1 Online Evaluation

In online evaluation, the system measures the change in user behavior when interacting with different recommender systems. In the case of rating prediction tasks, the value of such predictions can depend on a variety of factors such as the user's intent (e.g. information needs, novelty vs risk), the user's context (e.g. familiar items, system trust), and the interface through which the predictions are presented.

The advantages of online experiment are that the entire performance of the recommender can be evaluated such as long-term business profit and users' retention. Therefore, online experiment can be used to understand the impact of evaluation metrics on the overall performance of the system. The progressive evaluation process will reduce the risk of online experiment and accomplish satisfying recommendation results.

However, in a multitude of cases, such experiments are very costly, since creating online testing systems may require much effort. Furthermore, we would like to evaluate our algorithms before presenting their results to the users, in order to avoid a negative user experience for the test users. For example, a test system that provides irrelevant recommendations, may discourage the test users from using the real system ever again.

1.5.2 Offline Evaluation

The goal of the offline evaluation is to filter algorithms so that only the most promising need undergo expensive online tests. Thus, the data used for the offline evaluation should match as closely as possible the deployed online. Ensuring that there is no bias in the distribution of users, items and ratings, the experimenter may be tempted to prefilter the data by excluding non interesting items.

In order to evaluate algorithms offline, it is necessary to simulate the online process where the system makes predictions or recommendations, and the user corrects the predictions or uses the recommendations. This is usually done by recording historical user data, and then hiding some of these interactions in order to simulate the knowledge of how a user will rate an item, or which recommendations a user will act upon.

The advantage of offline analytics is that it doesn't need the interaction from real users, so it can be implemented at a low cost and can test and evaluate the performance of different kinds of recommendation algorithms quickly. But the disadvantages are such experiments can usually be used in evaluating the prediction (recommendation result) accuracy or precision of the algorithms.

The dataset is separated in a learning set having the recommendation results and a test set. The evaluation consists on comparing the proposed recommendations to the real existing results in the test set. Many techniques are used to split the dataset into training and test, for example we can use the cross validation technique to divide the dataset for k times (k=5 generally).

1.5.3 Evaluation metrics

Many evaluation metrics were proposed in order to measure the performance of recommender systems. Every metric is preferred in a specific kind of recommendation to give better results. Therefore, we are going to define the principle precision measurements in the case of ratings prediction and Top-N recommendation list. The fully detailed presentation of evaluation metrics is mentioned in the article [Shani and Gunawardana, 2011].

1.5.3.1 Precision metrics in vote prediction

The recommendation algorithms capable of predicting the rating of the items are the ones that utilized this precision metric. Noting that R is the set of votes r_{ui} of the users u from the set of users U on the items i from the set of items I, and to evaluate the recommender we divide the R on two distinct sets $R_{learning}$ and R_{test} where the first set is used for prediction $pred(u, i)$, and the second one is used to evaluate the precision of prediction results.

The set of couples (u, i) in the R_{test} where the recommender has already predicted their ratings is called T. The mainly used metrics to calculate the precision in the case of rating prediction recommenders are Mean Absolute Error (MAE) and Root Mean Squared Error(RMSE).

Mean Absolute Error (MAE): is used to evaluate the recommender system, the MAE calculates the mean absolute difference between the predicted ratings and the real ratings (in the R_{test}). The MAE equation is represented in the formula 1.5.

$$MAE = \frac{\sum_{(u,i)\in T} |pred(u,i) - r_{ui}|}{|T|} \tag{1.5}$$

Root Mean Squared Error (RMSE): is the most popular used metric in evaluating precision in recommendation. The RMSE equation is represented in the formula 1.6.

$$RMSE = \sqrt{\frac{\sum_{(u,i)\in T} (pred(u,i) - r_{ui})^2}{|T|}} \tag{1.6}$$

More the RMSE or the MAE value is lower, better the precision is. Of course, there are other metrics used to evaluate the precision like Normalized RMSE (NRMSE), Normalized MAE (NMAE) that utilize normalized values using a rating interval (vmin, vmax) and are applied on different datasets.

1.5.3.2 Precision metrics in Top-N list

If the recommender system does not calculate the rating values, the recommender returns a list of N items susceptible to interest the current user $L(u_a)$ which is named Top N recommendations. The precision measurement in the case of Top N recommendation shows if the item is pertinent to the current user or not. Similarly, the dataset is divided into $R_{learning}$ and R_{test}.

Precision and Recall: Precision and Recall are very popular evaluation measurements in Information Filtering Systems [Cremonesi et al., 2010]. They are widely used to evaluate Top N recommendation, I_{utest} is the subset of test items I_u present in the R_{test}, and $P(u) \subset I_{utest}$ is the pertinent list of items from the test base for the user u. Recall (respectively precision) is calculated for every user u then aggregated for all the users U.

The precision measures the proportion of the really pertinent recommendations for the user u from the list of recommended items $L(u)$. The formula 1.7 defines the precision measurement equation.

$$Precision = \frac{1}{|U|} \sum_{u \in U} \frac{|L(u) \cap P(u)|}{|L(u)|} \tag{1.7}$$

The recall measures the rate of pertinent recommendations from the set of really pertinent items for the user u from the list of pertinent items $P(u)$ (in the test base). The formula 1.8 defines the recall measurement equation.

$$Recall = \frac{1}{|U|} \sum_{u \in U} \frac{|L(u) \cap P(u)|}{|P(u)|} \tag{1.8}$$

Higher the precision and recall are, better the performance of the recommender system is. While comparing such recommendation algorithms, precision and recall must be used simultaneously because they are inversely proportional switch the N value of the list. Thus, when N increases, the precision and recall decreases and vice versa.

F1-measure: A third measurement metric is the F1-measure largely used and introduced to evaluate recommender systems, this metric is calculated by combining precision and recall values as presented in the formula 1.9.

$$F1 = \frac{2 \times Precision \times Rappel}{Precision + Rappel} \tag{1.9}$$

1.6 Conclusion

Recommender systems ultimate purpose is to predict to every user the list of items that will interest him using such several techniques. The recommendation of the items can be done using a non personalized algorithm or a personalized algorithm, the results of every recommendation are evaluated then in order to calculate the performance of every recommendation algorithm.

The chapter has introduced non personalized recommender systems in the section 1.1 and personalized recommender systems in the section 1.2. Then, it has detailed the challenges of the recommender systems in the section 1.3. Besides, it has focused on presenting the data collection process in recommendation in the section 1.4. Ending by detailing the evaluation of the recommender systems in the section 1.5.

Chapter 2

Smart Video

Smart Video [Chen et al., 2015] has received much attention in the last few decades because of the potential progress of technology where internet becomes widely used. Recently, Smart Video is used in almost all the fields due to the set of integrated functionalities and services beside the focus on analyzing the behaviour of users to facilitate their search.

So, the chapter will introduce the smart video streaming process in the section 2.1, then, it will present the most popular streaming platforms for smart video in the section 2.2, ending by emphasizing the various researches and works done about recommender systems for Smart Video field in the section 2.3.

2.1 Smart Video streaming process

With the increasing use of smart video, various platforms are developed for the aim of dropping off several video contents (live or on demand) [Juluri et al., 2015] to allow target users to consume the existing content of these streaming platforms and access what they want to visualize. Other than that, smart video streaming platforms integrate smart video players to track and report every action a user perform toward the video.

Streaming is the process of moving different types of data across communication channels without a drop in quality providing the ability to consume media over the internet. The media file played on the client device is stored remotely, and is transmitted a few seconds at time. Streaming enables easier access to multimedia resources integrated in different web-

based applications [Hartsell and Yuen, 2006].

Today, with a fast enough internet connection, we can stream high-definition movies and videos. Although, streaming videos require a faster method of transporting data than TCP/IP, which prioritizes reliability over speed. We can find several platforms and applications that allow video on demand streaming, real time or live streaming videos, or both [URL12].

2.1.1 Live streaming

Live streaming [Smith et al., 2013] represents the video content that are delivered or streamed via the internet at the same time they are happening like live concerts, live radio, video conferences, matches, interviews, etc. Undoubtedly, live streaming presents the new wave of digital communication, content promotion, and content consumption.

To make a video streaming process for live content, some features have to be mentioned in the provided services such as the support for a compatible player to ensure the consistent user experience on various devices. Live video streaming applications are basically user friendly. Live streaming gives content creators and audiences such benefits:

- **Low connectivity:** People need simply an internet connection with a connected device to visit the website and follow the live streaming event.
- **Real-time streaming:** The content is shared in real time and without any change into it giving people a sense of belonging to the moment.
- **Less-time production:** Live content is actually easier to produce than polished on Demand video. Live streaming does not require time for production.
- **High quality streaming:** With the advancement of the technical tools and the internet connection, the live broadcasting ensures a very high quality like UltraHD, 4K, 8K, etc.
- **Fastest Growing Industry:** The use of live video streaming becomes viral that's why the major streaming platforms integrates the live streaming content relating to the platform business.

2.1.2 Video on Demand (VoD)

The Video on Demand [Jenner, 2017] is defined when the event is recorded on a digital support and saved in a server then is accessible by internet users like video clips, movies trailers, etc. Video on demand is in many aspects, the exact opposite of the live streamed video. Regularly, video on demand is the traditional way of creating and sharing content.

Video on Demand gives users the opportunity to watch videos at any time and from any internet-connected device. It also provides the ability to watch tutorials, educative and entertaining contents, ... VoD is needed because content can tolerate delays in publishing and is always accessible. Some other benefits of VoD are:

- **Connectivity:** Due to internet connection, VoD content is always accessible, everyone can watch content whenever and wherever.
- **Variety:** VoD covers content that can be edited before it is distributed. Also, it is much easier to go viral with VoD than with live streaming.
- **Suitability:** This is when the live broadcasting is not mandatory and the content can tolerate delays in publishing VoD is suitable option.
- **Post-production:** Post-production is where the magic of editing happens. With editing, creators can affect everything from the storytelling to the pacing of the videos.
- **More creative tools:** VoD provides more opportunity for creativity than live streaming. Creators can use tools that don't work with real-time video content. And, they can use the time between the production and post-production to plan and execute.

2.2 Smart Video streaming platforms

Smart video streaming platforms [URL13] became very popular and prevalent so we can find various platforms that are capable of streaming live videos, on demand videos, or both live and on demand videos. Platforms for watching video contents are more favorable alternatives nowadays. Some of the most known platforms are presented as follows.

2.2.1 Netflix

Certainly, Netflix [URL14] is the most popular platform for online video streaming content due to its huge number of connected subscribers (over 193 million worldwide subscribers by July 2020 [URL15]) that interact with the numerous available items. Netflix also created its own original TV shows and movies to acquire consumers and enhance the Customer Relationship Management (CRM) [Gomez-Uribe and Hunt, 2015].

The power behind the success of Netflix is absolutely due to its integrated recommender system that aims to understand users' tastes in order to offer them content willing to interest them. In this context, we note the well known Netflix challenge that targeted to propose and present the most performing recommendation algorithm as detailed in this article [Bennett et al., 2007].

Whereas, due to the scalability problem of the proposed recommender system resulted from the potential increasing in the number of users and items in the year of 2008, the algorithm proposed in 2007 became not performing and obsolete. Thus, a factorisation matrix solution using Singular Value Decomposition (SVD) algorithm was proposed to compensate for the scalability issue [Zhou et al., 2008].

The Netflix company is heavily data driven and it lies in the middle of the internet and the storytelling. The main source of income comes from users' subscription fees allowing users to stream data from a wide range of existing videos anytime on a variety of internet connected devices. Besides, Netflix offers an easy platform with high quality streaming videos with the possibility of creating a list, sharing videos, etc.

2.2.2 Amazon Prime Video

Amazon Prime Video [URL16] is another popular video streaming platform with over 150 million subscribers in January 2020 [URL17]. The service offers a wide range of high-quality original videos as well as provides content from other resources. The creation of Amazon Prime account will automatically give access to the streaming library. The platform gives the chance to watch movies anytime and anywhere.

Identically to Netflix, Amazon Prime Video utilized data about users to build a profile for every user so to recommend him interesting items. Data about users are name, age, sex, preferred genres, etc. And the usage data of the interactions are view search, consultation history, likes, shares, favorite items, added to list, annotated items, etc. These data are used to feed the integrated recommender system of Amazon.

Amazon launched recommendations to help people to discover what they might not have found on their own. Since then, the algorithms has been spread in the Web, tweaked to help people to find videos to watch or news to read, challenged by other algorithms and techniques, and adapted to improve diversity and discovery, recency, time-sensitive or sequential items, and many other problems [Smith and Linden, 2017].

2.2.3 YouTube

YouTube [URL18] is a popular social media platform used for personal use and for business. The uploaded videos will show up on YouTube and in different search engines (e.g Google). YouTube is considered the earliest pioneer in the market and is absolutely the most used platform. YouTube provides on demand videos, live streaming videos and give access to publish user-generated content [Zhou et al., 2010].

The American online video-sharing platform allow users to upload, view, rate, share, add to playlists, report, comment on videos, and subscribe to other users. It offers a wide variety of user-generated and corporate media videos [Davidson et al., 2010]. Available content includes video clips, TV shows, music videos, documentary films, audio recordings, movie trailers, live streams, video blogging, and educational videos, etc.

YouTube offers a free space to the user to put any information about him with the possibility of connecting freely to the platform without even creating an account. Then the platform tracks the interactions of every user toward the published videos like the search and view history, annotations, ratings, shares, likes, reactions to the videos, etc. What makes enhances the performance of YouTube recommender system [Covington et al., 2016].

YouTube's recommender system learns from two types of user feedback: user profile

(clicks, watches, etc) and satisfaction behaviors (likes, dislikes). The recommender models the ranking problem as a « combination of classification and regression problems » with multiple objectives. The ranking model predicts the probabilities of the user which is a point-wise prediction model, rather than a pair-wise or list-wise [URL19].

2.2.4 Canal +

Canal + [URL20] is a French premium television channel launched in 1984 owned by the Canal+ Group, which in turn is owned by Vivendi. The channel broadcasts several kinds of programming, mostly encrypted. From the year 2005, Canal+ has integrated a more immersive personalized recommender system to give suggestions for every user. Canal+ is available to all internet users.

Meanwhile, on demand videos are films, series, sport videos, documentaries, etc. from Canal+, HBO, Fox, National Geographic, Paramount, History, Filmbox, BBC, TVN24 and CNN. Canal+ increases the discoverability of content in an intuitive way by displaying weighted and categorized keywords for every piece of content [URL21].

2.2.5 Other platforms

Many other platforms were developed to stream video content for all kind of users. These streaming platforms focus on streaming various content and especially enhancing the quality of the recommender system to give accurate recommendations that interest users.

Vimeo: is a high-quality service that offers a range of proprietary products for video streaming where users can discover videos from best creators around the world and share their own ones. Vimeo is powered with real-time analytics that provides detailed information on the streaming and audience engagement to help their constantly improvement. All the tools and features make Vimeo a useful tool for business live event streaming.

Kaltura: is one more well-known and widely used platform for online video sharing. The portal makes it easy to broadcast and share everything from staff training classes to lectures and academic conferences. Kaltura brings together contributors from various fields, gives a chance to like, comment, and share the liked feedback. In addition, the platform is empowered

with analytical tools that can provide useful insight data.

Hulu: is a video streaming platform that offers on demand content of classical and new TV shows and movies. Hulu launched the Live TV option that offers live streams of over 60 TV channels (broadcast and cable) where networks are CBS, ABC, NBC, Disney Channel, National Geographic etc. Hulu Live TV pleases the tastes of the wide audience as it offers news, channels, sports, and entertainment contents.

2.2.6 Smart TV

Smart TV or connected TV [Pan, 2017] is a television set with integrated internet and interactive Web 2.0 features which allow users to stream music and videos, browse the internet, and view photos. Smart TV is simply a smart television that can connect to internet and contain applications (installed by the user) such as Netflix, Amazon, Skype, YouTube, etc.

Beside the traditional functions of television provided through the traditional broadcasting of media, these devices can also provide Internet TV, home networking access, online interactive media, over-the-top (OTT) content as well as on-demand streaming media, etc. Smart TV can deliver content (such as photos, movies and music) from other computers or network attached storage devices on a network.

2.3 Smart Video and Recommender Systems

With the development and the expansion of TV programs and video contents, hundreds of internet-based content providers are available to users providing various multimedia contents such as television programs, movies, news, musics and so on. As a result, these smart video streaming platforms implement and apply very important recommender systems [Mei et al., 2009].

Focusing on recommender systems for smart video, we are going to give an overview of the recommendation concept in the field of smart video in the subsection 2.3.1, then we will give an overview or a survey of the recently realized work related to recommendation in smart video streaming platforms in the subsection 2.3.2.

2.3.1 Recommendation in Smart Video

Noting that integrating recommender systems into smart video platforms becomes a necessity to enhance the user experience and to comply with user needs while selecting and visualizing videos. Moreover, the crucial part of every successful and frequently used platform relies absolutely on the quality of the recommender system used to offer users the range of contents that fit their preferences [Ho et al., 2014].

Different from traditional services, smart video platforms greatly enrich the user interaction with the traditional platforms by providing the interactive video on demand service or Live service. So, the recommendation system should give appropriate suggestions of what users might like. This becomes an indispensable part for video streaming systems to acquire users and ensure their retention.

Smart streaming platforms allow the collection of implicit and explicit user preferences, such as user's playlists and behavior log, to make recommendation. Hence, famous television makers and content providers, such as Google TV, Apple TV, Sony TV, YouTube, Netflix, Hulu, and so on have concentrated on enhancing the performance results of their developed recommender systems.

2.3.2 Researches and work done in recommendation for Smart Video

Studying the existing research papers that focused on recommender systems in smart video platforms, we find a lot of existing work since the last few decades especially these recent years where the notion of smart videos and recommendations has improved exponentially due to the big amount of available data.

Thus, data is largely provided and can be easily exploited to serve the recommendation process and give higher results. A survey about the existing work was done using recommender systems in the field of smart video is represented in the following part.

A TV program recommender framework:

[Chang et al., 2013] The research paper proposed a smart and social TV program recommender framework for Smart TV and addresses several issues such as accuracy, diversity, nov-

elty, explanation and group recommendations which are important in building a TV program recommender system.

The smart and social TV program recommender framework consists of TV program content analysis, user profile analysis, and user preference learning module to process TV program content and to extract related information such as TV watching statistics information, users' preferences and interests for other contents from social media or relevant organization.

Automatic and personalized recommendation of TV program contents using sequential pattern mining for smart TV user interaction:

[Pyo et al., 2013] The paper proposed an automatic recommendation for TV program contents in sequence using Sequential Pattern Mining (SPM) [Fournier-Viger et al., 2017] based on TV viewer's behaviors of watching multiple TV program contents in a row.

A sequence of TV program contents were recommended to the target user from similar users. Three types of SPM methods were presented: offline, online and hybrid SPM. Then, to extract sequential patterns of preferably watched TV program contents, a Preference Weighted Normalized Modified Retrieval Rank (PW-NMRR) metric for similar user clustering [Micaelian et al., 2004] was used.

The offline SPM method is superior in relative long-sequence recommendation and the online SPM method was effective for short-sequence recommendation. The hybrid SPM method compromises its performance between the offline and online SPM methods.

Multi-modal Learning for Video Recommendation based on Mobile Application:

[Jia et al., 2015] The paper proposed a method to develop an Android background service to collect the users' behavior and analyze their preferences using a Multi-modal Generative Model based on textual descriptions of Android applications and videos (social services, gaming, health care, e-commerce, etc.) used by users.

These descriptions are used then to extract the video content based features to develop a personalized recommender system. The proposed and implemented recommendation algo-

rithm was an innovative model for video recommendation on cell phones for both long-term preferences and short-term habits in order to better simulate the users' judgements on relevance and interests.

Watch-It-Next: A Contextual TV Recommendation System:

[Aharon et al., 2015] This work addressed the recommendation challenge presented by multi-user Smart TV devices through leveraging the available context. The purpose of the paper is to propose a model to provide recommendations for the next program to watch after the currently watched program.

Then, it presented an empirical evaluation of several recommendation methods over large-scale real-life TV viewership data basing on the context to implicitly disambiguate users (specifically for household smart TV situation). The proposed recommender model utilizes Latent Factor Model [Jenatton et al., 2012] and Latent Dirichlet Allocation [Blei et al., 2003].

The proposed model outperformed the personalized non-contextual as well as contextual non-personalized baselines.

The video recommendation system based on DBN:

[Hongliang and Xiaona, 2015] This paper proposed a video recommendation system which combines Deep Belief Network (DBN) with Collaborative Filtering algorithm to achieve a better performance in accuracy compared with traditional content-based and collaborative filtering recommendation methods.

The research paper extended the classical depth of neural network model (DBN model) combined with User Based collaborative filtering algorithm to build video recommendation system splitting the user-item matrix into matrices used as the input of the DBN model to easily figure out the user profile and find the neighborhood relationships.

Then, the User-based Collaborative Filtering was deployed to calculate the missing remarks and find out the recommended movies for Netflix dataset.

Differentially Private Online Learning for Cloud-Based Video Recommendation with Multimedia Big Data in Social Networks:

[Zhou et al., 2016] The paper proposed a private learning framework for video recommendation for online social networks tackling with large volume of heterogeneous data concerning the privacy of social network users and video vendors.

The model utilized Exponential Mechanism and Laplace Mechanism [Xiaojian et al., 2014] simultaneously to design an efficient and high-accurate timely recommendation system based on multimedia cloud computing [Mell et al., 2011]. User-generated multimedia is translated to remote media cloud and stored in decentralized data centers.

Then the system extracted the user's context vectors to convert results and recommend video contents. The proposed model showed accurate results and performed a trade-off between performance loss and privacy preserving level.

A Context-Aware Method for Top-k Recommendation in Smart TV:

[Liu et al., 2016] The paper proposed an effective video Recommendation model for Smart TV service (RSTV) based on the developed Latent Dirichlet allocation (LDA) [Blei et al., 2003] to make personalized top-k video recommendations.

Besides, it presented proper solutions for some critical problems of smart TV recommender systems such as sparsity problem and contextual computing by using a real world dataset gathered from Hisense smart TV platform and JuHaoKan Video on Demand dataset (JHKVoD).

Recommendations were made based on associations between users past shown interests and context-aware post-filtering method (Contextual Computation). The proposed context-aware recommendation model outperformed other baselines in Recall, AUC, MAP and MRR.

Content-Based Video Recommendation System Based on Stylistic Visual Features:

[Deldjoo et al., 2016] This paper investigated the use of automatically extracted visual features of videos in the context of recommender systems and brings some novel contributions

by proposing a new content-based recommender system.

The proposed recommender system encompassed a technique to automatically analyze video contents and extract a set of representative stylistic features (lighting, color, and motion) grounded on existing approaches of Applied Media Theory.

The evaluation of these recommendations showed that applying traditional content-based recommendation techniques to exploit explicit content features lead to more accurate recommendations when visual features are extracted from full-length videos and movie trailers.

Also, the recommender addressed the problem originated from video files that have no metadata.

Enhancing recommender systems for TV by face recognition:

[De Pessemier et al., 2016] To automate user identification and feedback process for TV applications, the paper proposed a solution based on face detection and recognition services (Face++ and SkyBiometry). These services output useful information such as an estimation of the age, the gender, and the mood of the person.

Demographic characteristics (age and gender) were used to classify the user and cope with the limited user feedback and user identification which is called the cold start problem in recommendation. The system had an automatic identification process of the users sitting in front of the TV.

Also, the similarity indicator of the face recognition process was used to decide if a person already utilized the recommender system to deduce his preferences. The performance results of the proposed model showed accurate results in the case of frontal view of the face.

Videopedia: Lecture Video Recommendation for Educational Blogs Using Topic Modeling:

[Basu et al., 2016] This paper designed a system for integrating the blogs and the videos containing educational multimedia data and proposed a novel recommendation system called Videopedia.

The proposed recommender system integrated multiple media formats and automatically recommend relevant educational videos for blogs like Wikipedia. The model utilized topic modeling on automatically generated video transcripts from various video sharing platforms and used them as a representation of the video content.

The promising results showed that topic modeling is a good way of video recommendations. The integrated framework reduced the users efforts in searching relevant e-learning videos and provided a mechanism for content based search.

The Impact of Profile Coherence on Recommendation Performance for Shared Accounts on Smart TVs:

[Lian et al., 2017] The paper identified a novel characteristic profile coherence on smart TVs where an account is shared by multiple users. The model measured the coherence of users account's interests.

The coherence was computed as the average similarity between items in the account profile. Furthermore, it aimed to evaluate the impact of profile coherence on the quality of recommendation lists for coherent and incoherent accounts generated by different variants of item-based collaborative filtering.

Experiments conducted on a large-scale watch log on smart TVs conform that the profile coherence indeed impact the quality of recommendation lists giving better results in various aspects—accuracy, diversity and popularity especially in the case of applications where an account is shared by multiple users.

Context-aware media recommendations for smart devices:

[Otebolaku and Andrade, 2017] This article investigated the context-aware recommendation techniques for implicit delivery of contextually relevant online media items and proposed a recommendation module using a contextual user profile and context recognition framework.

The data determined from the user's current contextual preferences relied on a context

recognition service, which identifies user's dynamic contextual situation from device's built-in sensors. Besides, a mobile movie recommendation system is developed on top of the proposed system as a proof of concept to evaluate the proposed solution.

Experimental evaluation results showed how contextual information can influence the recommendation efficacy, helping to filter out preferences that are not relevant to the target user in the present context during filtering process.

Effective Knowledge Based Recommender System for Tailored Multiple Point of Interest Recommendation:

[Vijayakumar et al., 2019] The research paper presented a novel knowledge-based travel recommender system to help active target user by providing tailored travel recommendations incorporating the most efficient travel trajectory between one location to another based on crowd-sourced digital-footprints.

The model was employed on the mobile device to generate multiple point of interests based on heuristic search-based travel planning algorithm which organized locations based on time-specific relevance. The highly relevant point of interests was selected for recommendation as destinations.

Based on three different approaches such as Location Affinity, Hybrid Collaborative Filtering and Heuristic Search based Travel Planning Algorithm, the proposed travel recommender was experimentally evaluated with the real-time large-scale dataset and the results are accurate and proficient.

An Intelligent Video Tag Recommendation Method for Improving Video Popularity in Mobile Computing Environment:

[Zhou et al., 2019] This paper proposed a novel hybrid method based on multi-modal content analysis that recommended keywords for video uploaders to compose titles and tags of their videos and then to gain higher popularity.

The model focused on dealing with how to improve the popularity of video sharing on

websites. The model generated candidate keywords by integrating techniques of textual semantic analysis of original tags and recognition of video content.

Taking the original keywords as input to the Term Frequency Similarity (TF-SIM) to sort and rank the candidate keywords and recommend the most relevant keywords. The experimental results showed that the proposed method can effectively improve the social popularity of the videos.

Movie Genome Recommender: A Novel Recommender System Based on Multimedia Content:

[Deldjoo et al., 2019] The research paper presented a novel content centric web-based framework for movie search and recommendation powered by a pure Content Based Filtering (CBF) model exploiting rich semantic descriptors of movies' contents.

The movies descriptors included audiovisual features (state of the art audio and image descriptors) and metadata (genre and tags), which were called « Movie Genome » features, the model supported users with wide range of functionalities that were available on the vast majority of video on demand streaming services such as Netflix.

This method facilitated the execution of controlled empirical studies and serve the implementation of personalized recommendation algorithms. The performing Movie Genome Recommender was entirely web-based and can be run on a various devices.

Video on demand recommender system for internet protocol television service based on explicit information fusion:

[Seo et al., 2020] The research paper proposed a novel integration method of Internet Protocol television (IPTV) and Over-The-Top (OTT) for the IPTV Video on Demand VoD recommender system.

The proposed recommender aimed to extend explicit preference information between IPTV and (OTT) services to solve the data sparsity problem. The proposed IPTV VoD recommender algorithm was based on Probabilistic Factorization Matrix (PMF) that is more accurate

than memory-based recommender algorithms.

Furthermore, the proposed model was sufficiently flexible to guarantee high performance in most domains and to guarantee a high performance recommendation than these of the existing methods that used implicit preference to extend the explicit preference information for users in VoD contents.

Conclusion

Smart Video is certainly one of the most innovative fields that has been tackled by a huge number of researchers. Users consume the available streaming platforms and benefit from the intelligent aspect derived from using machine learning and recommendation algorithms. Thus, focusing on streaming various content to users and improving the recommendation results is crucial for the success of these platforms.

This chapter defined the streaming process of video content (Live and Video on Demand) in the section 2.1. Then, it presented the most popular video streaming platforms and how they focus on enhancing their recommender systems in the section 2.2. Ending by presenting the use of recommender systems in Smart Video streaming platforms mentioning the various researches and works done in this field in the section 2.3.

Chapter 3

Data Collection

xAPI standard [Bakharia et al., 2016] is a software specification developed for the e-learning field but has been adopted in several other fields due to the big success and popularity it has encountered because of its various advantages and benefits. xAPI is utilized to collect, store, and manage data about wide range of learning experiences or activities a person performed online and offline.

Besides, Experience API defines a standard used to represent data in a simple and well structured way. Moreover, xAPI is used to alleviate the heavy ETL process that needs a lot of time and treatment to extract data, transform them, and load them. So, thanks to xAPI, the ETL process is seen only as a simple data collection where data will be easily processed for the further tasks.

The chapter will introduce the SmartVideo project in the section 3.1 defining the platform and its functionalities. Then, we will present the xAPI standard focusing on the xAPI statement in the section 3.2. After, we will detail the xapi recipe elaborated for SmartVideo project in the section 3.3, ending by pinpointing the possible communication methods between xAPI and SmartVideo in the section 3.4.

3.1 SmartVideo project

SmartVideo Grand Est project (SVGE project) aims basically to provide a smart video streaming platform. The platform streams regional productions in France starting with Grand-

Est region. The SmartVideo project's main purpose is to focus on the regional audiovisual productions that were limited to local visualizations and valorize the regional culture by diffusing content for a wide range of users.

3.1.1 SmartVideo Grand Est platform

Thereby, the platform focuses on applying advanced techniques to provide an ergonomic, comfortable, and user friendly interface. It also integrates intelligence artificial methods to analyze users and know more about their preferences so to offer them personalized contents. The major potential of every streaming platform is the quality of the integrated recommender system.

For example, Netflix platform implement a strong and performing recommender system [Gomez-Uribe and Hunt, 2015] what makes the 80% of the viewed videos are issued from recommendation algorithms (in 2020 [URL22]). That's why, the platform worked on developing a powerful and strong recommender system that provides interesting recommendations willing to fit the users' tastes.

The recommender system's quality is based on the performance of the recommendation algorithms and the usage data collected about users, items, and their interactions. As a result, a profound research has been done to decide about the best way to represent data easily in consistent structure and how these data will be gathered and used in recommendation.

The Grand-Est platform permits diverse local televisions to put their video on demand and live contents to a large public. So, the platform allow users to find different contents for their favorite local channels and to discover new contents basing on recommendations. The user is asked to register to be able to connect to the platform and get some privileges.

SmartVideo is a non intrusive platform that respects the General Data Protection Regulation (GDPR) [URL23]. So, the user is asked only for his pseudo, email address, and eventually his village. The registration page is presented in the figure 3.1.

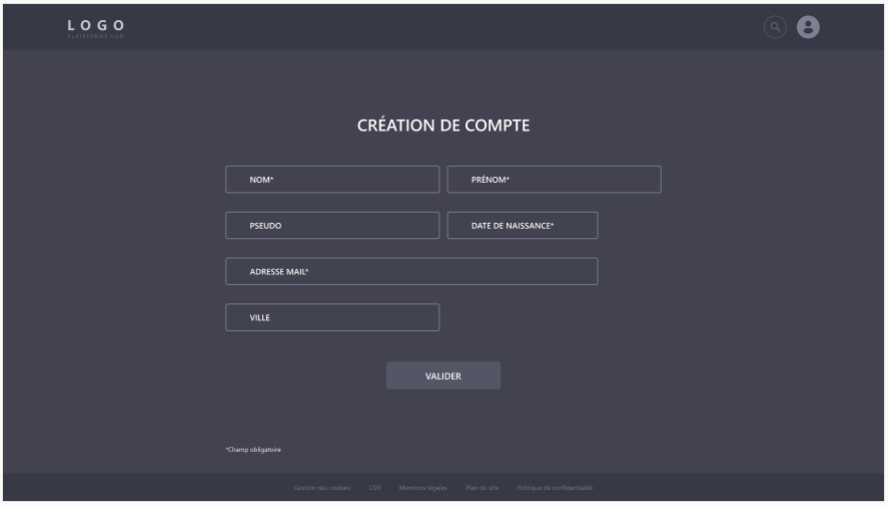


Figure 3.1: Registration page for a new user

3.1.2 Functionalities of the platform

Once the visitor of the platform has logged in, he can access the video content shown in the platform. The content is organized by the recommendation algorithm processed. The figure 3.2 shows some available content.

Figure 3.2: Videos organized by recommendation algorithms

Besides, the platform permits Grand-Est local televisions to put their contents. The Live streaming process is managed to make the user able to return back or to restart the streaming process from the beginning. The VoD content is diverse relating to channels' available thematics like actuality, politics, economy, society, sport, culture, entertainment, territory, patrimony, arts, etc. as shown in the figure 3.3.

Figure 3.3: Disposition by video thematic

The platform allow users to subscribe to certain channels and create their own playlists. The users can get notified by new interesting comings, leave their notations toward videos whether a rating (vote) or an annotation (comment or tag), or the possibility of liking or sharing a video on social media as presented in the figure 3.4.

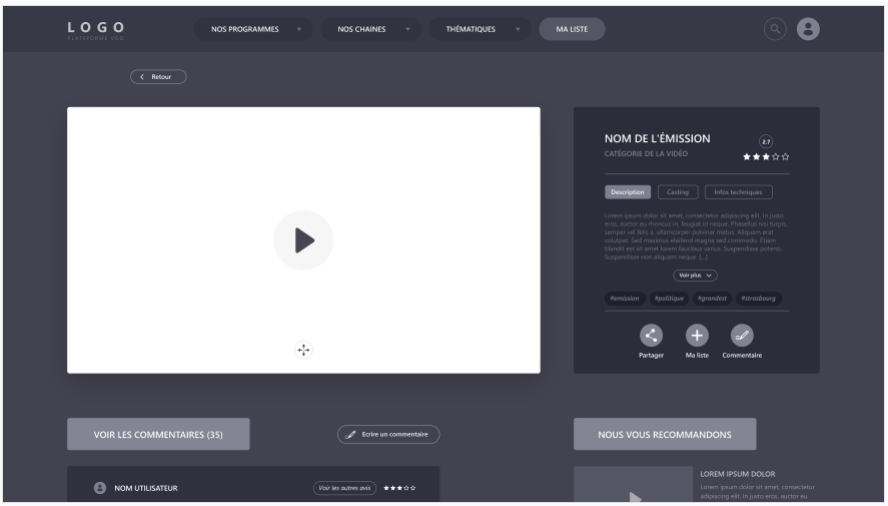


Figure 3.4: Possible interactions to the selected video

The platform provides personalized suggestions using different algorithms. The firstly introduced algorithms that the platform manage to develop and integrate are:

- **Because you liked:** The recommender system focuses on the time spent on consultations, the viewed content typologies, the ratings, the viewed programs, the stored videos in the playlist of the user, etc. to recommend similar videos the user will likely be interested in.
- **Popular:** The popular videos are presented by the number of views of the videos by users

with the visualisation duration. They are identical for all users but they can have a light personalization layer.

- **Newly:** The newly added items is a recommendation algorithm used to visualize the lastly added videos in the catalogue.
- **Nearby:** The nearby algorithm recommend the approximate videos which are geographically close to the users basing on their profiles' informations and IP addresses.

All the cited functionalities of the platform generate different data that ought to be tracked and analyzed in order to perform recommendations and provide interesting results. One ultimate challenge is the data representation and collection process. Of course, using clean, organized, and well structured data will facilitate the tasks of the recommender.

Also, data must be collected thoroughly without any altering or change. Therefore, we have adopted the xAPI standard, basically used in e-learning, and we have decided to adjust it for representing and storing the usage data as well as to gather these stored data easily. The purpose of adopting the xAPI standard is to facilitate the Data collection process.

3.2 xAPI Standard

Experience API offers a specific standard to represent data uniformly in a simple and well structured form [Serrano-Laguna et al., 2017]. Experience API is mainly a software specification implemented basically for the e-learning field to collect, store, and manage data about wide range of learning experiences.

3.2.1 Experience API in depth

The xAPI specification is a definitive document authored specifically for individuals and organizations to outline how learning experiences and learner activities can be tracked or recorded [Lim, 2015]. In 2011, Advanced Distributed Learning (ADL), the United States Department of Defense-sponsored stewards of SCORM, recognized the need for a newer and more capable software specification.

This new software was proposed to outperform SCORM (Shareable Content Object Reference Model) which has been the de facto e-learning standard for packaging e-learning content. xAPI was developed to address the issues and drawbacks of SCORM [Nouira et al., 2018]. The specification was developed with learning experiences in mind, but many other kinds of activities could be tracked using xAPI.

The xAPI specification [URL24] also defines some optional security methods allowing trusted exchange of information between trusted sources. Due to xAPI, different e-learning systems are able to securely communicate by capturing and sharing the stream of learning activities. The xAPI specification aims to maximize the interoperability of systems that create, gather, store, access, and retrieve data about learning experiences.

Furthermore, xAPI standard provides a guide to those who want to build conform applications to xAPI specification. xAPI is certainly the most versatile e-Learning standard adopted by almost all the e-learning platforms because it addresses various modern e-learning problems [Kevan and Ryan, 2016].

3.2.2 Benefits of xAPI standard

xAPI is considered as an empowering standard elaborated to alleviate a lot of issues [Berg et al., 2016]. Using xAPI standard will guarantee different benefits such as:

- **Data capture:** xAPI allows the capture of data from vast array of sources, both online and offline. Learners can interact with material on one device and pick up where they left off on another device later.
- **Integration:** The ability of xAPI to capture data from different sources and integrate them into one learning record to get a holistic view of the learning activities which identifies the learning interventions.
- **Data Transfer:** Data stored can be easily retrieved and transferred not only within the Learning Record System but also to other reporting systems. This guaranties the easy sharing of data.

- **Data security:** xAPI serves as a security method allowing the exchange of information between the Learning Record Store and trusted sources. Data is important and will often contain sensitive information.

- **Data structure:** xAPI also helps in the structuring and definition of state, learner, activity, and objects, which are means by which experiences are conveyed by a learner activity provider.

- **Data analytics:** Data can be very valuable, they can be used to review previous learning experiences, or analyzed to plan future projects by mapping what the learner knows against what they need to know.

- **Better insights:** xAPI can help getting better insights about learners by tracking their actions across multiple devices and environments pinpointing areas where users exit or have issues and what tutorials users interaction improves.

- **Learning assets:** Experience API gives powerful tools for analyzing the impact of learning. The analytics clearly show which parts need to be improved for future learners, which are working well, and areas in which the organization should focus on.

- **Flexibility:** xAPI allow learners access all kinds of material from any location (while travelling, doing jobs, or socializing with friends). xAPI allow tracking all the learning experiences in one simple consistent format.

When an learning experience is realized by a learner, this activity needs to be recorded. The application sends secure statements in the form of « Noun, verb, object » or « I did this » to be stored into a Learning Record Store (LRS).

3.2.3 Terminologies of xAPI standard

In this part, we are going to introduce and define the most used terminologies (keywords) in the xAPI standard basing on the GitHub specification URL [URL25] and on the article [Lim, 2016] .

- **Learning Experience:** An event associated with learning. Examples include reading a book, taking an online course, going on a field trip, engaging in self-directed research, or receiving a certificate for a completed course.

- **Activity Provider (AP):** Generates xAPI statements and an LRS Endpoint which receives statements and stores them in a database. An Activity Provider can be an LMS, a standalone course, video, game, simulator, a medical device, etc.
- **Statement or Learning Record:** A data structure consisting of <actor (learner)>, <verb>, <direct object>, with <result>, in <context> to track an aspect of a learning experience.
- **Learning Record Store (LRS):** A system that stores statements, generally a database. The xAPI is dependent to an LRS to function correctly. An LRS can store a wide range of learning experiences, real-world activities, mobile applications usage, or job performance.
- **Internationalized Resource Identifier (IRI):** A unique identifier used to identify an object such as a verb, activity or activity type.
- **Inverse Functional Identifier (IFI):** An identifier which is unique to a particular person or Group.
- **Learning Management System (LMS):** A software package used to administer one or more courses to one or more learners. An LMS is typically a web-based system that allow learners to authenticate themselves, register for courses, complete courses and take assessments.
- **Learning Record Consumer (LRC):** An xAPI Client that accesses data from the Learning Record Store(s) with the intent of processing data, including interpretation, analysis, translation, dissemination, and aggregation.
- **Learning Record Provider (LRP):** An xAPI Client that sends data to the Learning Record Store(s) to be stored for further processing task.

3.2.4 xAPI Statement

The experiences are recorded in the form of simple (actor, verb, object) statements (with result and context extensions) [Lim, 2018]. Also, documents and files can be attached. These statements are gathered by the Activity Providers and sent to the LRS. xAPI uses these statements to track data and report them back to a learning management system (LMS), Learning Record Store (LRS), or any application that understands the xAPI language.

A simple example of « actor, verb, object » statement is "Mary completed health training". A simple example of « actor, verb, object, result, context » statement is "John attempted a Test mission with 85% in Mathematics". Other examples are presented in the GitHub specification URL [URL26].

The figure 3.5 represents the full structure of an xAPI statement [URL27].

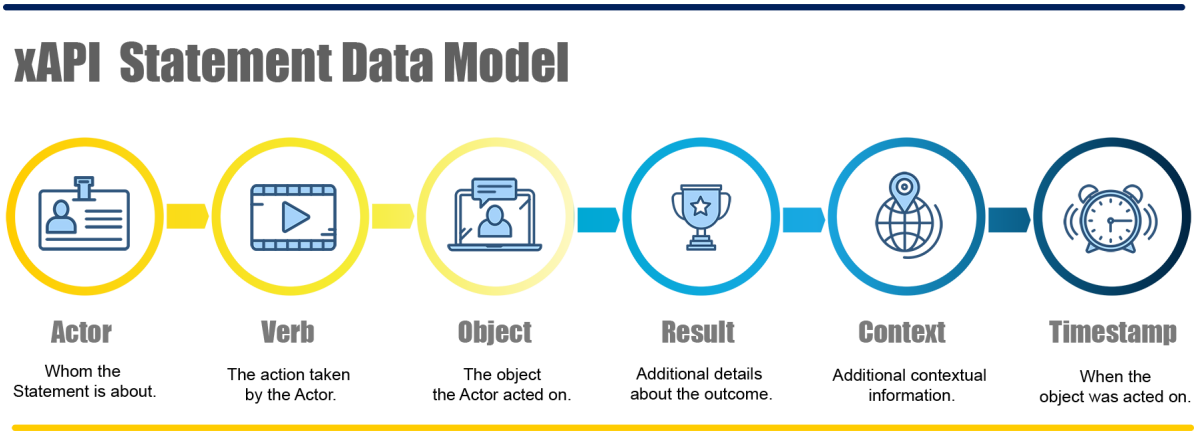


Figure 3.5: xAPI statement general structure
[URL28]

Technically, an xAPI statement is represented by a JSON[1] object that is transferred to the LRS via a REST request. The request could be GET, POST, PUT and DELETE and it can be performed only from an authorized client due xAPI security constraints [URL29]. The authorized client can be either a Learning Record Provider or a Learning Record Consumer that is identified by its username and password in the LRS [Bakharia et al., 2016].

The xAPI is not only a learning technology specification but also a suite of web-service APIs that support a simple object-based model for describing, recording, and accessing the learner's learning activities. Moreover, the SmartVideo platform is developed in the purpose of being conform to the xAPI standard.

[1]JavaScript Object Notation

3.3 xAPI for SmartVideo platform

Diving deeper on the usage data of SmartVideo Grand Est platform, when a visitor connects to the platform, he can obviously perform a set of actions. These actions are tracked and saved to serve the recommendation. The xAPI standard was adopted and tailored to represent all kind of activities or actions a user realizes while interacting with the platform.

Every interaction of the user to a video is described by the sentence « User did this ». This sentence is represented by a structured xAPI statement and is stored into a Learning Record Store (database). As a result, an xAPI video recipe or video profile [URL30] was thoroughly studied and elaborated to represent and depict every possible interaction of the users.

A recipe is a standard way of expressing a particular type of experience. It provides a common language and prevents the use of different words saying the same thing. Recipes allow different systems to talk about things people do using the same words. Basically, a recipe specifies which verbs, activity types, and extensions to use. Also, it promote consistency by using the same words [Bakharia et al., 2016].

After having an overview about the most popular video streaming platforms mentioned in the section 2.2 of the chapter Smart Video 2, a list of verbs is defined to describe the users' actions. Verbs are: Connected, Searched, Selected, Shared, Liked, Rated, Annotated, Added to playlist, Deleted from playlist, Played, Paused, Seeked, Completed, and Terminated. These listed 14 verbs are capable of presenting all the possible interactions of a user.

For every verb, an xAPI statement is associated. Noting that interactions may have extra informations like the user rated the video by 4/5, we managed to put these informations in the Context property of the xAPI statement for consistency rules. As a result, the general structure of our xAPI statements will contain the required Actor, Verb and Object properties, the indispensable Timestamp property and the optional property Context.

Our defined recipe provides an overall presentation about all kinds of performed interactions in a video streaming platform. Hence, we will adopt the xAPI statement structure and adjust it for our needed data. We are going to pinpoint how verbs, objects, and actors are iden-

tified using the xAPI specification [URL31].

Starting with verbs, we note that xAPI does not define semantic domains. Therefore, a list of repositories of verbs, objects, and extensions was already proposed to be utilized with the possibility of introducing new repositories. Hence, we are going to utilize the predefined ADL-NET[2] repository to present some existing verbs that we will be present in our xapi video recipe.

And, we will present our proper SmartVideo repository for the newly defined verbs. In fact, a **verb** is identified by an Internationalized Resource Identifier (IRI) and a display as presented in the table 3.1.

Display	**IRI Identifiant**
connected	`https://smartvideo.fr/xapi/verbs/connected`
searched	`https://smartvideo.fr/xapi/verbs/searched`
selected	`https://smartvideo.fr/xapi/verbs/selected`
shared	`https://smartvideo.fr/xapi/verbs/shared`
liked	`https://smartvideo.fr/xapi/verbs/liked`
rated	`https://smartvideo.fr/xapi/verbs/rated`
annotated	`https://smartvideo.fr/xapi/verbs/annotated`
added-to-playlist	`https://smartvideo.fr/xapi/verbs/added-to-playlist`
deleted-from-playlist	`https://smartvideo.fr/xapi/verbs/deleted-from-playlist`
played	`https://w3id.org/xapi/video/verbs/played`
paused	`https://w3id.org/xapi/video/verbs/paused`
seeked	`https://w3id.org/xapi/video/verbs/seeked`
completed	`https://w3id.org/xapi/video/verbs/completed`
terminated	`https://w3id.org/xapi/video/verbs/terminated`

Table 3.1: Description of xAPI verbs

The structure of the verb using xAPI is represented in the table 3.2.

[2]Advanced Distributed Learning Initiative

Property	Attribute	Value
Verb	IRI Identifiant	`https://SmartVideo.fr/xapi/verbs/liked`
	Display	liked

Table 3.2: Verb structure of xAPI statement

The **object** is described by its type (`Object Type`) which is always `Activity` and by its IRI identifiant using our proposed sub-domain following the form `https://SmartVideo.fr/xapi/objects/<videoID>`. The verbs (`Connected` and `Searched`) do not have an object so we attribute void to the IRI identifiant `https://SmartVideo.fr/xapi/objects/void`.

The structure of the object using xAPI is represented in the table 3.3.

Property	Attribute	Value
Object	IRI Identifiant	**No object:** `https://smartvideo.fr/xapi/ /objects/void` **Video object:** `https://smartvideo.fr/xapi/ /objects/<videoID>`
	Object Type	Activity

Table 3.3: Object structure of xAPI statement

The **actor** is represented by his type (`objectType`) that is always Agent, and by an Inverse Functional Identifier (IFI) `mailto:<userID>@SmartVideo.fr` containing his identifiant (userID). The structure of the actor is represented in the table 3.4.

Property	Attribute	Value
Actor	IRI Identifiant	`mailto:<userID>@smartvideo.fr`
	Type	Agent

Table 3.4: Actor structure of xAPI statement

The **Timestamp** is essential to trace the date and the hour of the activity. It is written

in the ISO 8601 format. If the xAPI statement does not contain the timestamp, the Learning Record Store (LRS) attributes automatically the date and hour when the statement is stored. And, the **Context** property is used to add the necessary extra informations of the activity.

In the following, we are going to pinpoint all the 14 statements as well as their properties (required and optional) with some specification values.

3.3.1 Connected Statement

If a user visits the platform for the first time, an authentication process takes part and if authentication successful, a new session is opened for the connected user and a Connected statement is sent to the LRS. Tracing the localisation of the user is recommended in order to serve some recommendation algorithms based on the geolocalisation.

If the user gives its localisation (latitude and longitude), the localisation is represented by a geolocalisation GeoJSON[URL32] object having the latitude and longitude coordinates. Else, the IP address of the user is reported by default.

The table 3.5 presents the xAPI statement corresponding to Connected verb.

Property	Attribute	Value
Actor	IRI Identifiant	`mailto:<userID>@smartvideo.fr`
	Type	Agent
Verb	IRI Identifiant	`https://SmartVideo.fr/xapi/verbs/connected`
	Display	connected
Object	IRI Identifiant	`https://SmartVideo.fr/xapi/objects/void`
	Object Type	Activity
Context	Geo JSON object or IP address	Latitude and Longitude values for Geo JSON object and IPV4 address for IP address
Timestamp		

Table 3.5: xAPI statement for Connected verb

3.3.2 Searched Statement

Searched statement describes the action of searching a video from the catalogue so the user must type the relative keywords to the corresponding video that will be saved in the context property. The table 3.6 presents the xAPI statement for Searched verb.

Property	Attribute	Value
Actor	IRI Identifiant	`mailto:<userID>@smartvideo.fr`
	Type	Agent
Verb	IRI Identifiant	`https://SmartVideo.fr/xapi/verbs/searched`
	Display	searched
Object	IRI Identifiant	`https://SmartVideo.fr/xapi/objects/void`
	Object Type	Activity
Context	Keywords	List of keywords
Timestamp		

Table 3.6: xAPI statement for Searched verb

3.3.3 Selected Statement

When the user select a video to watch, a Selected statement is formed to save the choice of the user. The selected video could be derived from a simple display, a search activity or a recommendation module. So, the Context property should track the source of this selection.

The table 3.7 describes the xAPI statement of Selected verb.

Property	Attribute	Value
Actor	IRI Identifiant	`mailto:<userID>@smartvideo.fr`
	Type	Agent
Verb	IRI Identifiant	`https://SmartVideo.fr/xapi/verbs/selected`
	Display	selected
Object	IRI Identifiant	`https://SmartVideo.fr/xapi/objects/<videoID>`
	Object Type	Activity

Context	Algorithm ID	Search or RecommendationAlgorithm-Name (i.e. CF-ItemBased, Basic-Popular, Basic-Recent)
	Rank	Rank of the video in recommendation
	Score	Score of the video in recommendation
Timestamp		

Table 3.7: xAPI statement for Selected verb

Remark: If the selection is not issued from a search activity or recommendation algorithm, the Context property is not required (extra).

3.3.4 Shared Statement

The activity of sharing a video on social media network such as Facebook, Instagram, Twitter, YouTube, ... is an important interaction that must be reported and the social media network is required in the Context property.

The table 3.8 describes the xAPI statement of Shared verb.

Property	Attribute	Value
Actor	IRI Identifiant	`mailto:<userID>@smartvideo.fr`
	Type	Agent
Verb	IRI Identifiant	`https://SmartVideo.fr/xapi/verbs/shared`
	Display	shared
Object	IRI Identifiant	`https://SmartVideo.fr/xapi/objects/<videoID>`
	Object Type	Activity
Context	Social Media ID	Network (i.e. Facebook, YouTube, WhatsApp)
Timestamp		

Table 3.8: xAPI statement for Shared verb

3.3.5 Liked Statement

The Liked statement describes the explicit evaluation of the user to a video. The like action could be True of False (Boolean) where True reflects the appreciation of the user to the video and False reflects his depreciation to the video. The xAPI statement should contain the appreciation or depreciation in the Context property.

The table 3.9 describes the xAPI statement of Liked verb.

Property	Attribute	Value
Actor	IRI Identifiant	`mailto:<userID>@smartvideo.fr`
	Type	Agent
Verb	IRI Identifiant	`https://SmartVideo.fr/xapi/verbs/liked`
	Display	liked
Object	IRI Identifiant	`https://SmartVideo.fr/xapi/objects/<videoID>`
	Object Type	Activity
Context	Value	True or False
Timestamp		

Table 3.9: xAPI statement for Liked verb

3.3.6 Rated Statement

If the user want to explicitly note a video, the rating is saved in the Context property under a JSON object named quality-rating containing the rating value and the range of min and max values for ratings. The table 3.10 shows the structure of the Rated statement.

Property	Attribute	Value
Actor	IRI Identifiant	`mailto:<userID>@smartvideo.fr`
	Type	Agent
Verb	IRI Identifiant	`https://SmartVideo.fr/xapi/verbs/rated`
	Display	rated

Object	IRI Identifiant	`https://SmartVideo.fr/xapi/objects/<videoID>`
	Object Type	Activity
Context	Raw	Rating value
	Min	Minimum value of ratings
	Max	Maximum value of ratings
Timestamp		

Table 3.10: xAPI statement for Rated verb

3.3.7 Annotated Statement

The user can eventually leave an annotation (tag or comment) to the video. This annotation must be traced into the statement to serve the recommendation. The table 3.11 describes the xAPI statement of Annotated verb.

Property	Attribute	Value
Actor	IRI Identifiant	`mailto:<userID>@smartvideo.fr`
	Type	Agent
Verb	IRI Identifiant	`https://SmartVideo.fr/xapi/verbs/annotated`
	Display	annotated
Object	IRI Identifiant	`https://SmartVideo.fr/xapi/objects/<videoID>`
	Object Type	Activity
Context	Comment	Annotation
Timestamp		

Table 3.11: xAPI statement for Annotated verb

3.3.8 Added to playlist Statement

The user can create his own list of videos and manage it by adding or removing videos. The Context property should save the playlist ID and the playlist name. The table 3.12 describes

the xAPI statement of Added-to-playlist verb.

Property	Attribute	Value
Actor	IRI Identifiant	`mailto:<userID>@smartvideo.fr`
	Type	Agent
Verb	IRI Identifiant	`https://SmartVideo.fr/xapi/verbs/added-to-playlist`
	Display	added-to-playlist
Object	IRI Identifiant	`https://SmartVideo.fr/xapi/objects/<videoID>`
	Object Type	Activity
Context	Playlist ID	ID of the playlist
	Playlist Name	Name of the playlist
Timestamp		

Table 3.12: xAPI statement for Added-to-playlist verb

3.3.9 Deleted from playlist Statement

If the user removes a video from his playlist, the action is directly sent to the LRS to be stored. The Context property contains the playlist ID to identify the playlist. The table 3.13 describes the xAPI statement of Deleted-from-playlist verb.

Property	Attribute	Value
Actor	IRI Identifiant	`mailto:<userID>@smartvideo.fr`
	Type	Agent
Verb	IRI Identifiant	`https://SmartVideo.fr/xapi/verbs/deleted-from-playlist`
	Display	deleted-from-playlist
Object	IRI Identifiant	`https://SmartVideo.fr/xapi/objects/<videoID>`
	Object Type	Activity

Context	Playlist ID	ID of the playlist
Timestamp		

Table 3.13: xAPI statement for Deleted-from-playlist verb

3.3.10 Played Statement

The Played Statement describes the player's start-up, the **Context** property is **optional** and is dependant to the ability of the player to track the visualisation period (duration) of the video. If the duration is traced, it is represented by the ISO 8601 format due to xAPI specification.

The table 3.14 describes the xAPI statement of Played verb.

Property	Attribute	Value
Actor	IRI Identifiant	`mailto:<userID>@smartvideo.fr`
	Type	Agent
Verb	IRI Identifiant	`https://w3id.org/xapi/video/verbs/played`
	Display	played
Object	IRI Identifiant	`https://SmartVideo.fr/xapi/objects/<videoID>`
	Object Type	Activity
Context	Duration	Duration of video visualization
Timestamp		

Table 3.14: xAPI statement for Played verb

3.3.11 Paused Statement

If the user clicks on the pause button, the statement is reported having a position value (in the Context property) presented in ISO 8601 format. The position value represents the duration of the video visualization from where the user begins till the pause action. The table 3.15 describes the xAPI statement of Paused verb.

Property	Attribute	Value
Actor	IRI Identifiant	`mailto:<userID>@smartvideo.fr`
	Type	Agent
Verb	IRI Identifiant	`https://w3id.org/xapi/video/verbs/paused`
	Display	paused
Object	IRI Identifiant	`https://SmartVideo.fr/xapi/objects/<videoID>`
	Object Type	Activity
Context	Position	Position in the video from where the user starts
Timestamp		

Table 3.15: xAPI statement for Paused verb

3.3.12 Seeked Statement

When the user changes the progression bar forward and backward, a Seeked statement having the start position and ending position (following the ISO 8601) format is reported. The table 3.16 describes the xAPI statement of Seeked verb.

Property	Attribute	Value
Actor	IRI Identifiant	`mailto:<userID>@smartvideo.fr`
	Type	Agent
Verb	IRI Identifiant	`https://w3id.org/xapi/video/verbs/seeked`
	Display	seeked
Object	IRI Identifiant	`https://SmartVideo.fr/xapi/objects/<videoID>`
	Object Type	Activity
Context	Starting Position	Position where the user sets the progress bar from
	Ending Position	Position where the user sets the progress bar to
Timestamp		

Table 3.16: xAPI statement for Seeked verb

3.3.13 Completed Statement

The completed statement depicts the completion of the video visualisation by the user. The table 3.17 describes the xAPI statement of Completed verb.

Property	Attribute	Value
Actor	IRI Identifiant	`mailto:<userID>@smartvideo.fr`
	Type	Agent
Verb	IRI Identifiant	`https://w3id.org/xapi/video/verbs/completed`
	Display	completed
Object	IRI Identifiant	`https://SmartVideo.fr/xapi/objects/<videoID>`
	Object Type	Activity
Timestamp		

Table 3.17: xAPI statement for Completed verb

3.3.14 Terminated Statement

The terminated statement describes the action of stopping the video playing process. The Context property is required to show the position (in ISO 8601) of the user while terminating the video. The table 3.18 describes the xAPI statement of Terminated verb.

Property	Attribute	Value
Actor	IRI Identifiant	`mailto:<userID>@smartvideo.fr`
	Type	Agent
Verb	IRI Identifiant	`https://w3id.org/xapi/video/verbs/terminated`
	Display	terminated
Object	IRI Identifiant	`https://SmartVideo.fr/xapi/objects/<videoID>`
	Object Type	Activity

Context	Position	Position in the video from where the user starts
Timestamp		

Table 3.18: xAPI statement for Terminated verb

3.4 Communication methods between xAPI and SmartVideo

At this stadium, we have to focus on the communication process between xAPI and the SmartVideo platform. Thanks to xAPI standard, the whole ETL process is transformed basically to a data collection process. So, the platform collects data from the LRS and uses these data to apply recommendation algorithms that will be used to recommend interesting content for every user.

3.4.1 Communication between xAPI and SmartVideo

Data are defined using xAPI specification and stored in a Learning Record Store separately. Then, gathered and processed to run recommendations. Noting that the recommender system developed for SmartVideo platform is considered as an independent black box containing the set of implemented recommendation algorithms that are executed using endpoints, the platform has to communicate with the LRS and the recommendation box to get results.

So, SmartVideo project can be globally seen as three communicating components that are the platform, the LRS, and the recommendation engine. The platform communicates with the LRS to gather data about interactions and usage, then sends the collected data to the recommender to apply recommendations. Results are transferred to the platform to personalize the visualized content of every user.

The figure 3.6 details the workflow of SmartVideo project and how the three components communicate with each other to gather data, apply recommendations, and send results to the platform.

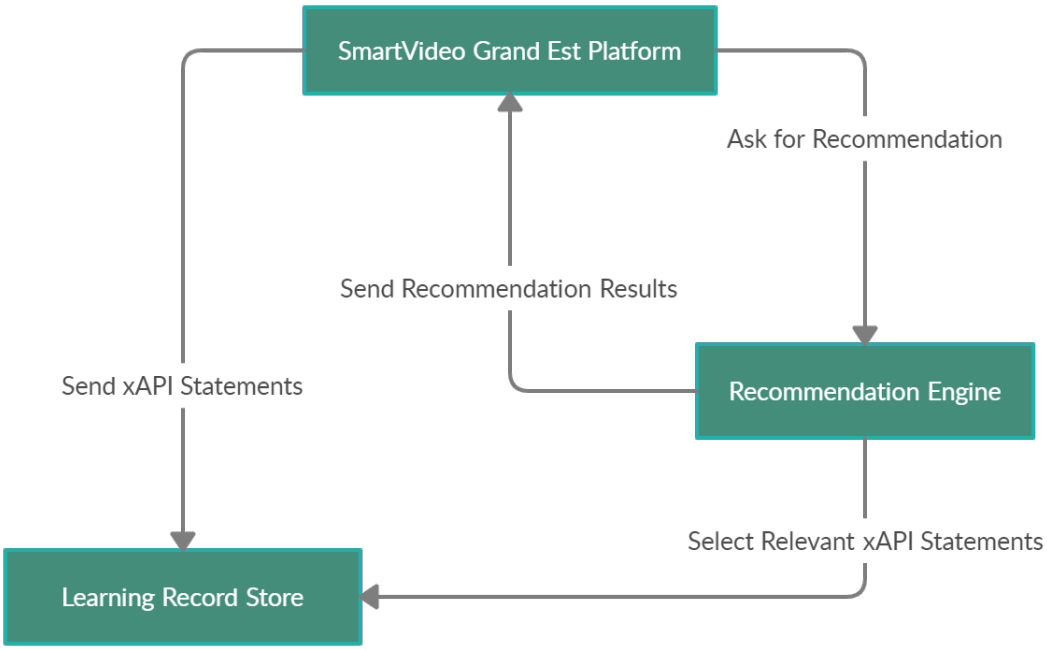


Figure 3.6: Communication workflow for SmartVideo project

The first step of the workflow is to represent data using the xAPI standard and store data into a Learning Record Store to facilitate their collection and manipulation. A lot of LRSs have been developed and used to store data (xAPI statements, profile activities, etc.). Thus, we have managed to utilize Trax LRS to store and deal with the stored xAPI statements.

3.4.2 Trax Learning Record Store

There is a bunch of existing Learning Record Stores [URL33] that can be open source or commercial such as Grassblade LRS [URL34], Watershed LRS [URL35], Yet Analytics [URL36], Trax LRS [URL37], and Learning Locker [URL38]. Of course, the functionalities offered by the different LRSs differ from the type of the LRS (open source or commercial).

For our case, we opted for using an open source LRS for better understanding and experiencing the provided functionalities pf the LRS. The only available open source LRSs are Learning Locker and Trax LRS. Researches proved that the use of Trax LRS is more convenient for the case of using xAPI video recipes [Labba et al., 2020, Labba et al., 2019].

That's why, we managed to utilize Trax LRS which is a free and open source Learning

Record Store that focuses on storing and managing data conforming to the xAPI standard. Trax LRS is developer-friendly and relies on a well known technology stack. It also permits working with Relational Databases, NoSQL Databases, or Hybrid Databases. Trax is a progressive LRS where data providers can add features, packages, functionalities, etc.

Trax LRS collects and tracks the learning experiences conforming to the xAPI standard. Besides, it focuses on storing and managing data basing on the latest xAPI specification. Trax LRS offers an xAPI compliant API as well as a secured User Interface accessed by an authentication system. xAPI is truly a community-driven specification with contributors participating from around the world.

Before being used, Trax LRS has to be installed. The installation guide is published on GitHub [URL39] to help users install the LRS correctly with all the needed dependencies. Other than that, adopting Trax LRS makes it possible to create proper packages and use them personally or even publish them to be used by all other users.

3.4.3 Data Collection Workflow

This section highlights the Data Collection Workflow for SmartVideo platform and xAPI. The workflow is used to gather data from Trax LRS and send them to the recommender system for applying recommendations. Data are exchanged between xAPI and SmartVideo platform using one of the three defined communication methods:

- The platform sends xAPI statements periodically to be stored in the LRS.
- The platform executes predefined methods to create xAPI statements and send them to the LRS.
- The platform uses the implemented endpoints used to create, store, and manage xAPI statements.

The data providers (Learning Record Providers) and the data consumers (Learning Record Consumers) must authorize to be able to manage statements.

3.4.3.1 Communication using xAPI statements

The platform can send xAPI statements every period (day, week, month). The xAPI statements are created manually by the data providers where every statement describes the interaction of the user in the platform. This is basically the most simple method whereas it takes a huge time to create statements manually.

3.4.3.2 Communication using predefined methods

We can also define a bunch of methods that aim to form the corresponding xAPI statements from the provided information. For every verb, we attributed a method that takes required and optional attributes as input, then build the correspondent xAPI statement.

- **connected_function**(Required userID: String, Optional localisation: GeoJSON Object, Default IP: IPV4, Optional timestamp: Timestamp);
- **searched_function**(Required userID: String, Required keywords: List<String>, Optional timestamp: Timestamp);
- **selected_function**(Required userID: String, Required videoID: String, Required algorithmID: String, Optional rank: Integer, Optional score: Float, Optional timestamp: Timestamp);
- **shared_function**(Required userID: String, Required videoID: String, Required socialMediaID: String, Optional timestamp: Timestamp);
- **liked_function**(Required userID: String, Required videoID: String, Required value: Boolean, Optional timestamp: Timestamp);
- **rated_function**(Required userID: String, Required videoID: String, Required raw: Float, Required min: Integer, Required max: Integer, Optional timestamp: Timestamp);
- **annotated_function**(Required userID: String, Required videoID: String, Required comment: String, Optional timestamp: Timestamp);
- **addedToPlaylist_function**(Required userID: String, Required videoID: String, Required playlistID: String, Required playlistName: String, Optional timestamp: Timestamp);

- **deletedFromPlaylist_function**(Required userID: String, Required videoID: String, Required playlistID: String, Optional timestamp: Timestamp);
- **played_function**(Required userID: String, Required videoID: String, Optional duration: ISO 8601, Optional, timestamp: Timestamp);
- **paused_function**(Required userID: String, Required videoID: String, Optional position: ISO 8601, Optional timestamp: Timestamp);
- **seeked_function**(Required userID: String, Required videoID: String, Required startingPosition: ISO 8601, Required endingPosition: ISO 8601, Optional timestamp: Timestamp);
- **completed_function**(Required userID: String, Required videoID: String, Optional timestamp: Timestamp);
- **terminated_function**(Required userID: String, Required videoID: String, Required position: ISO 8601, Optional timestamp: Timestamp);

3.4.3.3 SmartVideo package for Trax LRS

Because Trax LRS is open source and gives the possibility of adding new packages containing functionalities and services. We tried to facilitate the tasks of the data providers by developing our proper package that will allow managing xAPI statements easily.

The developed methods and endpoints of all the defined statements in the xAPI video recipe are presented in the table 3.19.

Endpoint	Method
`http://loria.kd-serveur.com/connection`	GET
	POST
`http://loria.kd-serveur.com/connection/<id>`	GET
	PUT
	DELETE
`http://loria.kd-serveur.com/search`	GET
	POST

http://loria.kd-serveur.com/search/<id>	GET
	PUT
	DELETE
http://loria.kd-serveur.com/selection	GET
	POST
http://loria.kd-serveur.com/selection/<id>	GET
	PUT
	DELETE
http://loria.kd-serveur.com/sharing	GET
	POST
http://loria.kd-serveur.com/sharing/<id>	GET
	PUT
	DELETE
http://loria.kd-serveur.com/like	GET
	POST
http://loria.kd-serveur.com/like/<id>	GET
	PUT
	DELETE
http://loria.kd-serveur.com/rating	GET
	POST
http://loria.kd-serveur.com/rating/<id>	GET
	PUT
	DELETE
http://loria.kd-serveur.com/annotation	GET
	POST
http://loria.kd-serveur.com/annotation/<id>	GET
	PUT
	DELETE
http://loria.kd-serveur.com/addition	GET
	POST

http://loria.kd-serveur.com/addition/<id>	GET
	PUT
	DELETE
http://loria.kd-serveur.com/deletion	GET
	POST
http://loria.kd-serveur.com/deletion/<id>	GET
	PUT
	DELETE
http://loria.kd-serveur.com/play	GET
	POST
http://loria.kd-serveur.com/play/<id>	GET
	PUT
	DELETE
http://loria.kd-serveur.com/pause	GET
	POST
http://loria.kd-serveur.com/pause/<id>	GET
	PUT
	DELETE
http://loria.kd-serveur.com/seeking	GET
	POST
http://loria.kd-serveur.com/seeking/<id>	GET
	PUT
	DELETE
http://loria.kd-serveur.com/completion	GET
	POST
http://loria.kd-serveur.com/completion/<id>	GET
	PUT
	DELETE
http://loria.kd-serveur.com/termination	GET
	POST

http://loria.kd-serveur.com/termination/<id>	GET
	PUT
	DELETE
http://loria.kd-serveur.com/statements	GET
http://loria.kd-serveur.com/statements/<id>	GET
	DELETE

Table 3.19: Endpoints and methods of SmartVideo Package

Conclusion

In guise of conclusion, the chapter introduces the SmartVideo Grand Est project emphasizing its architecture containing three main components that are relatively independent where the three components, which are the LRS, the SmartVideo platform, and the recommendation engine cooperate one with another in order to make recommendations.

Furthermore, the use of an xAPI standard is very recommended in this case of study in order to solve several problems faced in the Data Collection task. Moreover, xAPI is used for describing, storing, managing, and analyzing data that are conform to the xAPI specification and not the LRS, so any migration from an LRS to another will not effect the data nor require added tasks.

The chapter introduced SmartVideo project in the section 3.1 by defining the platform and its functionalities. Then, presented the xAPI standard in the section 3.2. Also, it detailed the xAPI recipe elaborated for SmartVideo project in the section 3.3, ending by giving the possible communication methods between xAPI and SmartVideo in the section 3.4.

Chapter 4

Recommendation algorithms

The developed recommender system integrated into our platform will contain a bunch of non personalized and personalized recommendation algorithms with the aim of testing and evaluating these algorithms using the corresponding evaluations metrics. Then, we will interpret the found results in order to decide which is the better recommendation algorithm for our case with our provided data.

The chapter will present the used dataset in the section 4.1. Then, it will introduce the used technologies and tool for realizing our work in the section 4.2. After, it will define the non personalized recommendation algorithms in the section 4.3 with their evaluations in the section 4.4. And, it will present the personalized recommendation algorithms in the section 4.5 with their evaluations in the section 4.6. Ending by a discussing results in the section 4.7.

4.1 Datasets

The SmartVideo Grand Est Regional project for France was already detailed in the section 3.1. Noting that the platform is actually in the development process. And, in order to test our developed algorithms, our collaborators have provided us data derived from an ancient platform. The provided data describe the navigation history of the users on some regional local channels like Via Vosges [URL40] and Alsace20 [URL41].

After analyzing these provided data, we have decided to keep working with the data related to the Alsace20 channel because Via Vosges channel focuses basically on streaming

news and actualities (recent contents). Whereas, Alsace20 channel streams various content such as movies, documentaries, actualities, news, shows, concerts, retransmissions, etc.

Description of Alsace20 Dataset:

The Alsace20 dataset provides 14 157 different videos, with 1 095 766 users, and about 5 813 497 different interactions between the users and the videos from the year of 2014 to the year of 2020. The provided data are categorized into data about users, data about items (videos), and data about interactions (usage data). From the given data, we are going only to utilize the ones that we will need in our case of study.

- **Data about users:** The provided data describing the users is the user ID.
- **Data about items:** The provided data about the items are the videoID, title, publication date, total duration, and mode (Live or Video on Demand).
- **Usage data:** The usage data provided are the visualization duration of the video and the timestamp (datetime when the interaction has happened).

Obviously, the provided data are massive in volume and in order to deal with this huge number of users, videos, and interactions we will need a lot of resources (time and space). That's why, we have managed to utilize only interactions of the year 2019 for our first experimental cases. Besides, we have decided to filter our data to get only the users that have made at least 5 positive visualizations (every visualization > 0).

So, the finally dataset contains 3192 different users and 3691 different videos. These users have processed about 30 559 video visualizations. However, we do not have any information about the ratings associated to the videos. That's why, we can not experiment the Most Popular algorithm based on the ratings (Best Rated and Most Rated) neither evaluate the recommendation results. Experimentations of rating based algorithms will be reported for further time.

4.2 Realization

In this section, we are going to present the frameworks, technologies, tools, programming languages, and libraries that we have utilized in order to fulfill our work.

4.2.1 Laravel Framework

Laravel [URL42] is a free and open source PHP framework that is robust and easy to understand. Laravel is widely used by web developers because it provide a modular packaging system with the possibility of creating own packages, an Object-relational Mapping (ORM), an automatic testing, etc. Trax LRS (section 3.4.2) was developed using Laravel application so we have developed our SmartVideo package (section 3.4.3.3) on the source code of Trax using Laravel architecture.

4.2.2 Flask Python Framework

Flask [URL43] is a micro web framework written in Python that does not require particular tools or libraries. It is a popular web framework, meaning it is a third-party Python library used for developing web applications. Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

4.2.3 Jupiter Notebook

The Jupyter Notebook [URL44] is an open-source web application that allow creating and sharing documents containing live code, equations, visualizations and narrative text. Jupiter Notebook is a tool executing python scripts to apply data cleaning and transformation, numerical simulations, statistical modeling, data visualization, machine learning methods, and much more tasks.

4.2.4 OpenAPI/Swagger

The OpenAPI Specification [URL45], originally known as Swagger Specification, is a specification for machine-readable interface files for designing, building, documenting, producing, consuming, and visualizing RESTful web services. OpenAPI Specification (formerly Swagger Specification) is an API description format for REST APIs that supports various programming languages and allow generating clients and servers to execute the defined APIs.

4.2.5 GnuPlot

GnuPlot [URL46] is a portable command-line driven graphing utility that can be used in many platforms (Windows, Unix,etc). It was originally created to allow scientists and students to visualize mathematical functions and data interactively. But, GnuPlot has grown to support many non-interactive uses such as web scripting. It is also used as a plotting engine to plot 2D and 3D graphs using an ergonomic layout.

4.2.6 Programming Languages

The programming languages that we have used are PHP and Python.

4.2.6.1 PHP

PHP [URL47] is a popular general-purpose scripting language especially suited for web development. PHP is fast, flexible and pragmatic. It can be used for many programming tasks outside of the web context, such as standalone graphical applications and robotic drone control. PHP was used to develop the SmartVideo package of Trax LRS using Trax LRS source code that is a Laravel application.

4.2.6.2 Python

Python [URL48] is an interpreted, high-level and general-purpose programming language that helps programmers to write clear logical code for small and large-scale projects. It is dynamically typed and garbage-collected. And, it supports multiple programming paradigms, object-oriented, and functional programming. We have used Python programming language for developing and testing our proposed recommendation algorithms.

4.2.7 Libraries

Using Python for developing our recommendation algorithms, we are going to introduce the used libraries that we have utilized in order to apply recommendations and evaluations for our case of study.

4.2.7.1 Surprise

Surprise [URL49] is a Python library for building and analyzing recommendation algorithms by providing a collection of estimators or prediction algorithms. It also supports tools for model evaluation like cross validation iterators and built-in metrics, as well as tools for model selection and automatic hyper-parameter search. Thanks to Surprise, users can implement their own recommendations with a minimal amount of code.

4.2.7.2 Pandas

Pandas [URL50] stands for "Python Data Analysis Library" which is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables, called dataframes, and time series. Pandas provide a lot of methods that can be performed in order to clean, process, transform and analyze data.

4.3 Non personalized recommendation algorithms

The recommendation resulted for the non personalized algorithms are common for all users. Although, we have decided to customize these recommendation results for every user by adding a simple **personalization layer** that consists of recommending only videos that the current user has **not visualized** yet. Thus, every user will see a customized list of recommended videos that are susceptible to interest him and that he has not watches already.

To detail the non personalized algorithms implemented for the platform, we have defined a set of annotations that tend to represent the used data in a formal and comprehensive way.

- u: User
- u_a: Current user
- U: Set of Users
- i: Item (Video)
- d_i: Total duration of the item i
- U_i: Set of users that have visualized the item i

- I_u: Set of items visualised by the user u
- $r_{u,i}$: Rating of the user u to the item i
- R_i: Set of ratings attributed to the item i

In the following, we are going to present the proposed and developed Most Popular and Most Recent algorithms.

4.3.1 Most Recent Algorithm

The Most Recent algorithm aims to filter the videos by their publication date (release date). So, when a new video is added to the platform, it is saved in order to be used for recommendation. Executing the Top N Most Recent algorithm for the current user u_a, a list of the top N recently added videos not seen by the current user u_a is displayed.

$$Recent\_Videos = Last(Release\_date) \tag{4.1}$$

4.3.2 Most Popular Algorithm

The Most Popular algorithm relies on filtering videos basing on several criteria. The Most Popular algorithm can be used to recommend the most viewed videos, the frequently viewed videos, the best rated videos, the most rated videos, etc. Therefore, we have decided to enrich our Most Popular algorithm using a list of criteria. In general, the criteria used depend on the visualizations (views) or the ratings.

So, we have categorized the algorithm to View Based Most Popular approach and Rating Based Most Popular approach. The View Based Most Popular approach contains the Most Popular algorithms basing on the absolute duration view, the relative duration view, or the view number. While, the Rating Based Most Popular approach contains the most rated and the best rated algorithms.

4.3.2.1 View Based Most Popular

The View Based Most Popular category focuses on the visualization (view) activity. So, when a user visualizes a video, an xAPI statement is created and stored to describe this ac-

tion. The visualization action can be described by the Played, Paused, Seeked, Completed, and Terminated statements containing the view duration value of the video.

4.3.2.1.1 Absolute Duration View: Analyzing the different visits of every user u, we will keep only the maximum absolute duration view spent on every video i. So, for every couple user-video, we will find one only the absolute duration view value that corresponds to the maximum absolute view duration.

$ad_{u,i,d}$ = The absolute duration view spent on the video i by the user u in seconds at a datetime d during one visit on the platform.

$ad_{u,i}$ is the maximum absolute duration spent by a given user u to a given video i in seconds during the different visits on the platform.

$$ad_{u,i} = max(ad_{u,i,d}) \tag{4.2}$$

The total absolute duration view of a video i by all the users is calculated using the formula 4.3:

$$ad_i = \sum_{u \in U} ad_{u,i} \tag{4.3}$$

The algorithm calculates the total durations spent on every video by all the users, then, it recommends the most viewed videos having the highest absolute duration views. Then, every user will get only the videos that he hasn't watched yet.

4.3.2.1.2 Relative Duration View: The used videos have different durations d_i between 2 minutes and 3 hours. So, if a user visualizes 1 minute from a video having 2 minutes, so, the user have visualized 50% of the video. And, if the user watch 5 minutes from a video having 120 minutes duration, so he has watched only 4% of the video. Thus, we decided to use the relative view duration to deal with this diversity in durations.

The Relative Duration View is a normalized value between 0 and 1 that corresponds to the average of visualisation of a given user to a given video.

$rd_{u,i}$ = The relative view duration of the video i by the user u in seconds.

$$rd_{u,i} = ad_{u,i}/d_i \tag{4.4}$$

The total relative duration view of a video i by all the users is calculated through the formula 4.5:

$$rd_i = \sum_{u \in U} rd_{u,i} \tag{4.5}$$

The algorithm calculates the total relative durations spent on every video by all the users. Then, it recommends the highly viewed videos having the highest relative duration values. Every user will get the videos that he has not watched before.

4.3.2.1.3 View Number: The occurrences number of the visualizations performed by a user to a video. The algorithm counts the number of visits of a video by the set of users. Then, the recommendation will give for every user the most frequently viewed videos that he has not watched yet.

The total number of views of a video i by all the users U is presented in the formula 4.6:

$$vn_i = |U_i| \tag{4.6}$$

4.3.2.2 Rating Based Most Popular

The Rating Based Most Popular category utilizes the rating values associated to the videos by the users to apply recommendations. The ratings can be generated implicitly or explicitly, and they are normalized between 0 and 1.

4.3.2.2.1 Best Rated: The highest ratings of the users to the videos. The Best Rated algorithm filters the videos by their ratings to get only the highly rated videos (videos having more than 0.5 rating value). And, the recommender will use these highly rated videos to recommend the highly rated videos not visualized by the current user u_a.

$hr_{u,i}$ = Rating of the user u to the video i having a normalized rating value grater or equal 0.5.

$$hr_{u,i} = r_{u,i}\ Where\ r_{u,i} >= 0.5 \tag{4.7}$$

The Best Rating of the video i is calculated by dividing the sum of ratings of the highly rated videos by the number of all the highly rated videos.

HR_i = Set of high ratings (>= 0.5) attributed to the video i

$$br_i = \frac{\sum_{u \in U} hr_{u,i}}{|HR_i|} \tag{4.8}$$

4.3.2.2.2 Most Rated: The algorithm utilizes the number of ratings assigned by the different users to the videos and applies recommendation to display for every user the list of Top N most rated videos that he did not watch before. The Most Rated videos are calculated by counting the number of all the ratings of the users to every video.

$$mr_i = |R_i| \tag{4.9}$$

4.4 Evaluation of Non Personalized algorithms

The experimental phase consists of introducing the evaluation results of the developed non personalized algorithms following by a discussion about these results. Getting inspired from the prediction and Hit Rate metrics [Schein et al., 2002] used basically to evaluate the Top N recommendation algorithms, we opted for proposing a more consistent evaluation algorithm that fits exactly our case of study. Our proposed evaluation algorithm aims to:

1. Filter dataset by the visits of the users to the items from February 2019 to December 2019 (leaving the data traced in January 2019 for the cold start problem). The set of resulted visits is noted by V.

2. For every visit $v(u, i, t)$ of the user u to the video i at a given time t, get the current user and the current video.

3. Keep the current video i in the test and process recommendation for the current user u on the previous period ($period < t$).

4. Compare whether the removed video (in the test set) exists in the list of Top N recommendation or not. If yes, $success(v) = 1$. Else, $success(v) = 0$.

5. Once all the visits are looped, the precision score is calculated using the formula 4.10 where v represents the visit $v(u, i, t)$.

$$Precision = \frac{\sum_{v \in V} success(v)}{|V|} \tag{4.10}$$

6. Repeat the process of calculating the hit rate prediction for the different N numbers of top N recommendations where $N \in \{5, 10, 15, 20\}$.

4.4.1 Evaluation Results

At this stadium, we are supposed to show the precision scores and deduce the best non personalized recommendation algorithm for our data. The figure 4.1 illustrates the hit rate precision scores of all the tested algorithms.

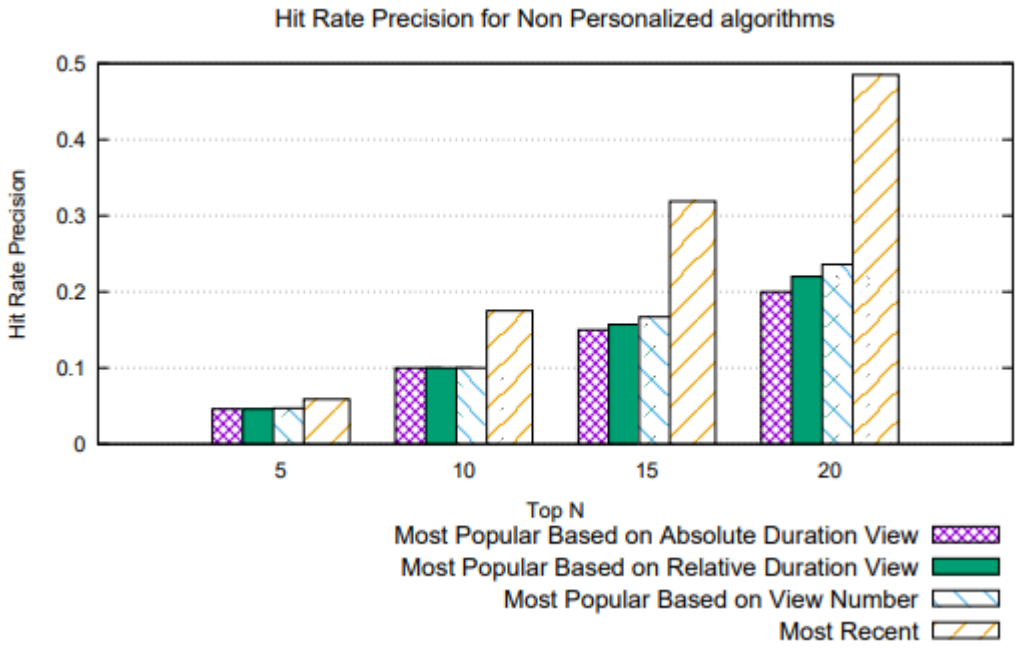


Figure 4.1: Hit Rate Precision for Non personalized Recommendation algorithms

The figure 4.1 shows that the Hit Rate Precision for the Most Recent and Most Popular algorithms varies between 0.05 and 0.5 depending on the used criterion and the top N number of recommendation.

4.4.2 Results Analysis

Diving deeper to explain the precision results found in the figure 4.1, we have deduced some interpretations such as:

- Varying the number of Top N influence the recommendation results. So, the higher the number of Top N is, the better the hit rate precision results are.
- The Absolute Duration View and Relative duration View have produced the same precision scores so calculating the Relative Duration View is not needed for our case because it requires more computational time than using the Absolute Duration View.
- The Most Popular based on the View Number shows better precision results than the Absolute Duration and Relative Duration based algorithms, so we do not need using the duration because it requires computational time and complexity than using the simple and more performing View Number based algorithm.
- The Most Recent algorithm shows the highest **precision** score up to **0.485** for $N = 20$ which is certainly justified by the fact that the usage data provided from the local channel Alsace20 present majorly content is about news and actualities. So, if our usage data focuses more on the movies, the Most Popular algorithm will probably show better results.

4.5 Personalized recommendation algorithms

We are going to highlight the personalized recommendation algorithms that we are going to implement which are the Memory Based and Model Based approaches of the Collaborative Filtering technique. CF algorithms can be performed using many kind of evaluations such as purchases, views, clicks, likes, etc. Thus, we will utilize the visualization task (view) to apply recommendation because we do not dispose of ratings.

The Collaborative Filtering recommendation technique was introduced and defined generally in the section 1.2.2.1. CF bases on two main approaches: Memory Based and Model Based. The Memory Based approach has two categories which are Used Based and Item Based recommendations. Whereas, the Model Based approach consists of applying Machine Learning predefined models to make recommendations.

4.5.1 Memory Based Collaborative Filtering

The common point between the User Based and Item Based approaches of the Memory Based Collaborative Filtering technique is the determination of the neighbors of the user u in the case of user based algorithms or the determination of the neighbors of the item i in the case of item based algorithms. Determining neighbors is processed by measuring the similarity of the user (resp. the item) with the potential candidates.

Therefore, we are going to define how to calculate the similarity between the users (resp. the items) for the case of User Based (resp. Item based) approach. Then, we will underline how the neighbors are determined after calculating similarities.

4.5.1.1 Similarity calculation

The similarity calculation aims to determine the neighbors of a user (or an item). In the neighbors' based algorithms as presented in this section, the similarity between two users is calculated by measuring the existing correlation between their visualizations (views). The similarity between two users u and w can be measured using the Pearson correlation coefficient, Cosinus coefficient, Jaccard coefficient, etc.

We note $sim(u, w)$ as the function that measures the similarity between two users u and w (resp. $sim(i, j)$ that measures the similarity between two items i and j). We also define $I_{uw} = I_u \cap I_w$ as the set of rated items by both users u and w, (equivalently, $U_{ij} = U_i \cap U_j$ is the set of users that have rated i and j items). The $neighbors(u)$ is the set of users $w \in U$ defined as the neighbors of the user u.

- **Cosinus Coefficient:** A measure of similarity between two users very utilized in Information Search and Filtering [Salton et al., 1997]. In the case of collaborative filtering, every user u is represented by a vector $x_u \in R^{|I|}$, where $x_{u,i} = v_{u,i}$ if the user u has evaluated the item i. Giving the missing values, the cosinus is calculated on the set of items viewed by the two users applying the formula 4.11.

$$sim(u, w) = \cos\left(\overrightarrow{\mathrm{x_u}}, \overrightarrow{\mathrm{x_w}}\right) = \frac{\sum_{i \in I_{uw}} v_{u,i} \times v_{w,i}}{\sqrt{\sum_{i \in I_{uw}} v_{u,i}^2} \sqrt{\sum_{i \in I_{uw}} v_{w,i}^2}} \tag{4.11}$$

The Cosinus varies between 0 and 1. A value equal to 1 indicates that the two users have identical preferences, and a value equal to 0 indicates that the two users have nothing in common. The major inconvenient of using cosinus in collaborative filtering that cosinus does not take into account the variation in the users' judgements.

- **Jaccard Coefficient:** A classic measure of similarity [Niwattanakul et al., 2013] between two sets. Given the set of visualized items I_u and I_w by the users u and w, Jaccard only counts the number of common evaluations (visualizations) over the size of the union of sets I_u and I_w (number of unique elements) [Thada and Jaglan, 2013] as presented in the formula 4.12.

$$sim(u,w) = Jaccard(u,w) = \frac{\mid I_u \cap I_w \mid}{\mid I_u \cup I_w \mid} \tag{4.12}$$

The Jaccard varies between 0 and 1. A value equal to 1 indicates that the two users have identical preferences, and a value equal to 0 indicates that the two users do not have common preferences. The Jaccard coefficient is majorly used in the case of values between 0 and 1.

4.5.1.2 Selection of neighbors

A first selection of the neighbors of the user u (resp. the item i) consists on ignoring all the negative similarities. The higher positive similarity shows a good indicator to prove that the two users (resp. the two items) share the same group of interests. While, a negative similarity shows that they share different interests.

Experimentations reported in the literature proved that the prediction relevance is sensible to the k number of nearest neighbors [Spiliopoulou et al., 2003, Sarwar et al., 2001] and is following a concave curve. When the k number is low (k < 10), the prediction relevance is low. And, when k increases, the number of the neighbors contributing to the prediction increases as a result the prediction relevance increases.

Finally, the pertinence decreases if a lot of neighbors participate in the prediction (k > 60), it is due to the fact that the strong relationships with the nearest neighbors are diluted by the weaker ones.

4.5.1.3 User Based Prediction

The user based prediction was introduced for the first time by the GroupLens system [Resnick et al., 1994] with the following functioning principle:

- Determining the current user's neighbors by calculating his similarity with other users.
- Calculating the visualisation prediction of the current user u_a on the item $i \in I$ by analyzing the visualisations of the neighbors of u_a on the same item i.

4.5.1.3.1 Calculating Similarity: The similarity is calculated using a Similarity Coefficient (Cosine or Jaccard) as presented in the section 4.5.1.1. Hence, the list of the current user's neighbors is determined as explained in the section 4.5.1.2.

4.5.1.3.2 Calculating Prediction: To calculate the prediction of the visualization of the current user u_a to the video i, we weight the view duration of every neighbor by his similarity value with the current user in order to deal with the diversity of users' distances relating to the current user [Ticha, 2015].

The visualisations of the neighbors will have most important weights than the furthest neighbors. Noting that the sum of weights of all the neighbors is not equal to 1, and in order to normalize the prediction values, the sum is divided by the sum of similarities of the current user by his neighbors. The calculation of the prediction is calculated applying the formula 4.13.

$$pred(u_a, i) = \frac{\sum_{u \in neighbors(u_a) \cap U_i} sim(u_a, u) v_{u,i}}{\sum_{u \in neighbors(u_a) \cap U_i} |sim(u_a, u)|} \tag{4.13}$$

In the initial algorithm implemented in GroupLens system, all the neighbors are taken into account while calculating the prediction. Besides, using the k nearest neighbors ameliorates not only the prediction relevance but also the efficiency of the algorithm [Schafer et al., 2007].

4.5.1.4 Top N User Based Recommendation

Top N recommendation algorithms are generally utilized when the system does not contain numeric values described by a scale of values. The available values can be binary or unary. As for the user based prediction, the first step consists of defining the nearest neighbors of the

current user u_a using a similarity coefficient (sections 4.5.1.1 and 4.5.1.2).

Then, the second step consists of determining for every neighbor w, the list L_w of the pertinent items (visualized, purchased). The items are then sorted using the frequency of their presence in the list of all the neighbors. And, the Top N most frequent items will be recommended to u_a [Sarwar et al., 2000a].

4.5.1.5 Item Based Prediction

The Item Based approach was introduced by [Sarwar et al., 2001], it is processed to make recommendations basing on the items' similarities and neighborhood. The prediction of the user u to the candidate item $i \in I \backslash I_u$ is calculated from the visualizations of the neighbors of i. The functioning principle is:

1. For every candidate item i, we determine the nearest neighbors of the item by calculating its similarities with other items.
2. Calculate the prediction of the current user's visualisation for the candidate item i after analyzing the visualisations of the nearest neighbors of i.

4.5.1.5.1 Calculating Similarity: The similarity between two items is calculated using a Similarity Coefficient (Cosine or Jaccard) as presented in the section 4.5.1.1. Hence, the list of the nearest neighbors of the candidate item i is determined as explained in the section 4.5.1.2.

4.5.1.5.2 Calculating Prediction: To calculate the prediction of the visualization of the current user u_a to the candidate video i, we weight the view duration of every neighbor of the item i by his similarity value with the current item [Ticha, 2015].

Researchers [Sarwar et al., 2001] have demonstrated that the prediction relevance is very sensible to the number of considered neighbors. So, only the k nearest neighbors will be taken into account while calculating the prediction.

Noting that the sum of weights of the k nearest neighbors is not equal to 1, and in order to normalize the prediction values, the sum is divided by the sum of similarities of the current item i by his neighbors. The calculation of the prediction is calculated using the formula 4.14.

$$pred(u_a, i) = \frac{\sum_{j \in neighbors_k(i) \cap I_{u_a}} sim(i,j) v_{u,j}}{\sum_{j \in neighbors_k(i) \cap I_{u_a}} |sim(i,j)|} \tag{4.14}$$

The Item Based algorithms are less sensible to the missing values issue and are more performing in terms of efficiency (complexity) than the User Based algorithms. Although, experimentations have shown that the User Based algorithms are more performing in terms of precision than the Item Based algorithms [Deshpande and Karypis, 2004].

4.5.1.6 Top N Item Based Recommendation

The Top N item based recommendation algorithm [Deshpande and Karypis, 2004] consists of determining the neighbors of every item $i \in I$ not visualized by the current user u_a. The determination of the neighbors is processed by calculating similarities as presented in the section 4.5.1.1, then, the nearest neighbors were selected as detailed in the section 4.5.1.2.

The second part is to recommend for the current user the list of top N items [Ticha, 2015]. Given I_{u_a}, the list of items viewed by the current user u_a, for every item $i \in I \backslash I_{u_a}$ candidate, we calculate the sum of its similarities with its k nearest neighbors ($neighbors_k(i)$) which are visualized by the current user as presented in the formula 4.15.

$$x_{u_a}(i) = \sum_{j \in I_{u_a} \cap neighbors_k(i)} sim(j,i) \tag{4.15}$$

The list of Top N items to recommend to the current user u_a is given from the items having the highest similarity values $x_{u_a}(i)$.

4.5.2 Model Based Collaborative Filtering

The main drawback of Memory Based technique is the requirement of loading a large amount of inline memory. The problem is serious when the used matrix becomes so huge in situation that there are extremely many persons using the system. Computational resources are consumed much and the system performance goes down [Shani et al., 2005].

Model Based approach intends to solve such problems. In other words, we extract some information from the dataset, and use that as a « Model » to make recommendations without having to use the complete dataset every time. This approach potentially offers the benefits of

both speed and scalability.

Focusing on the Matrix factorization approach that takes advantages of the used matrix with regard to matrix algebra. Concretely, Matrix Factorization Based Collaborative Filtering aims to reduce the dimension of the matrix [Sarwar et al., 2000b] and to discover potential features under this matrix. These features will serve a purpose of recommendation [Do et al., 2010].

There are some models of matrix factorization in context of CF such as Latent Semantic Analysis (LSA) [Evangelopoulos et al., 2012], Latent Dirichlet Allocation (LDA) [Krestel et al., 2009], Principle Component Analysis (PCA) [Costello and Osborne, 2005], and Singular Value Decomposition (SVD) [Sarwar et al., 2002]. Thus, We are going to detail the SVD approach then we will execute and test the algorithm.

4.5.2.1 Singular Value Decomposition (SVD)

SVD is a method from linear algebra [Sarwar et al., 2002] that has been generally used as a dimensionality reduction technique in machine learning. SVD is a matrix factorisation technique [Koren et al., 2009] that reduces the number of features of a dataset by reducing the space dimension from N-dimension to K-dimension (where $K < N$) [De Lathauwer et al., 2000].

In the context of the recommender systems, SVD is used as a collaborative filtering technique. It uses a matrix structure where each row represents a user, and each column represents an item. The elements of this matrix are the evaluations assigned to items by users. The factorisation of this matrix is done by the singular value decomposition [Wall et al., 2003].

SVD is commonly used for producing *low-rank* approximations. Given an $m \times n$ matrix X, with rank r, the singular value decomposition, $SVD(X)$, is defined in the formula 4.16.

$$SVD(X) = U \times S \times V^T \tag{4.16}$$

Where U, S and V are of dimensions $m \times m$, $m \times n$, and $n \times n$, respectively. The Matrix S is a diagonal matrix having only r non zero entries, which makes the effective dimensions of these three matrices $m \times r$, $r \times r$, and $r \times n$, respectively. The figure 4.2 shows the Singular

Value Decomposition of the matrix X processed in the formula 4.16.

$$\underset{m \times n}{\overset{X}{\begin{pmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & \\ \vdots & \vdots & \ddots & \\ x_{m1} & & & x_{mn} \end{pmatrix}}} = \underset{m \times r}{\overset{U}{\begin{pmatrix} u_{11} & \dots & u_{1r} \\ \vdots & \ddots & \\ u_{m1} & & u_{mr} \end{pmatrix}}} \underset{r \times r}{\overset{S}{\begin{pmatrix} s_{11} & 0 & \dots \\ 0 & \ddots & \\ \vdots & & s_{rr} \end{pmatrix}}} \underset{r \times n}{\overset{V^{\mathsf{T}}}{\begin{pmatrix} v_{11} & \dots & v_{1n} \\ \vdots & \ddots & \\ v_{r1} & & v_{rn} \end{pmatrix}}}$$

Figure 4.2: Singular Value Decomposition of a Matrix

[URL51]

S is a diagonal matrix, called the Singular Matrix. The diagonal entries $(s_1, s_2, ..., s_r)$ of S have the property that $s_i > 0$ and $s_1 \geqslant s_2 \geqslant ... \geqslant s_r$. In general, we retain only $k \ll r$ singular values by discarding other entries. We term this reduced matrix S_k. Since the entries in S are sorted, the reduction process is performed by retaining the first k singular values. The matrices U and V are also reduced to produce U_k and V_k matrices, respectively.

Researchers [Deerwester et al., 1990, Berry et al., 1995] pointed out that the *low-rank* approximation of the original space is better than the original space itself due to filtering out of the small singular values that introduce « noise » in the user-item relationship.

4.5.2.2 Prediction generation using SVD

Given that the SVD somehow reduces the dimensionality of the dataset and captures the « features » that we can use to compare users. The first step is to represent the dataset as a matrix where the users are rows, videos are columns, and the individual entries are the visualisations.

Once the $m \times n$ rating matrix X is decomposed and reduced into three SVD component matrices with k features U_k, S_k, and V_k, we can predict the rating by simply looking up to the entry for the appropriate user-video pair in the resulted matrix $\hat{X}$. Prediction task can be generated by computing the cosine similarity (dot items) between m pseudo-users $U_k \cdot \sqrt{S_k}^T$ and n pseudo-items $\sqrt{S_k} \cdot V_k^T$ [Berry et al., 1995].

In particular, the prediction score $P_{i,j}$ for the i-th user on the j-th item is processed by adding the row average $\overline{r_i}$ to the similarity. Formally, $P_{i,j} = \overline{r_i} + U_k \cdot \sqrt{S_k}^T(i) \cdot \sqrt{S_k} \cdot V_k^T(j)$.

Once the SVD is done, the prediction generation process involves only a dot item computation, which takes $O(1)$ time, since k is constant.

4.6 Evaluating Personalized algorithms

The experimental phase consists of introducing the evaluation results of the personalized algorithms following by a discussion about these results. From our used dataset, we will utilize the Binary View and the Relative Duration values. We are going to store these values into two matrices: Binary View Matrix and Relative Duration Matrix (like the Rating Matrix).

- **Binary View Matrix:** contains in the rows the users, in the columns the videos, and in every case the binary value describing weather the user u has watched the video i or not (0: not watched, 1: watched).
- **Relative Duration Matrix:** contains the average of the visualization duration of the video i by the user u (0: not watched, 1: totally watched). The matrix contains the users in the rows, the videos in the columns, and the Relative Duration View $rd_{u,i}$ (defined in the section 4.3.2.1.2) in every case.

The evaluation metrics of the algorithms basing on the Binary View and the Relative Duration differ. In fact, the Binary View is evaluated using the precision score because we deal with binary values (0 and 1) so we can only execute a Top N recommendation list . While, the Relative Duration is evaluated using the RMSE because we utilize the relative duration values that predicts the visualization duration (average duration).

4.6.1 Binary View Based Evaluation

To evaluate our algorithms, we have split our dataset (Binary View Matrix) to train set and test set. We have applied the recommendation on the train set, then, we have compared the results with the test set. We have used the cross validation to divide the whole dataset into 5 folds where one fold was used for the test and the other four folds were used for the train. The purpose of the cross validation is to give exact precision by browsing the whole dataset.

For the case of Memory Based algorithms, we have dealt with the number k of nearest neighbors beside dealing with the similarity coefficients. So, we have tested the algorithms for

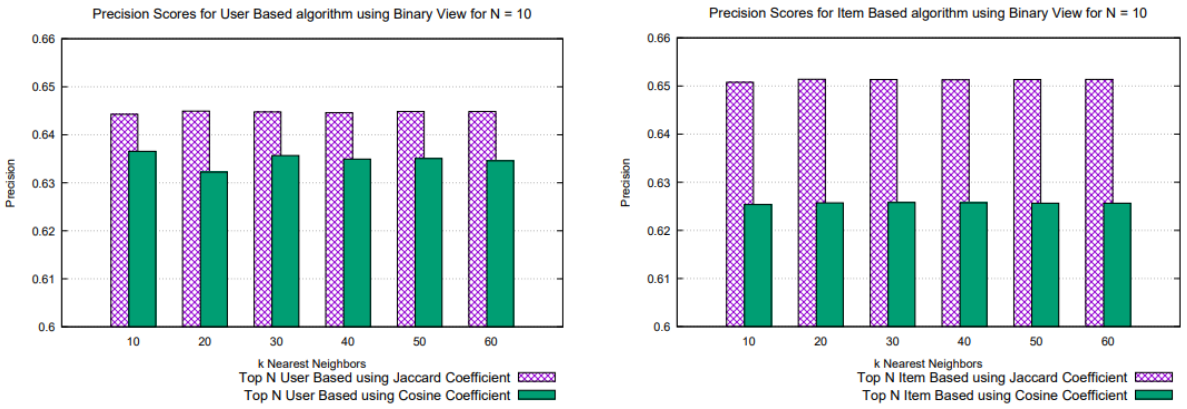
the case of $k \in \{10, 20, 30, 40, 50, 60\}$ using Jaccard similarity and Cosine similarity. The evaluations have pinpointed the best number of nearest neighbors (best k) and the best similarity coefficient used leading to the highest precision.

For the case of Model Based, we have processed by tuning the hyper parameters of the used SVD model and calculate the precision. Then, after getting the best precision value corresponding to the best hyper parameter configuration, we have applied the model to get predictions and recommendations for the current user u_a. In the evaluations, we have fixed the number of top N recommendation to 10 ($TopN = 10$).

Also, we have fixed a **threshold** of **0.5** that corresponds to the average view of the video (video watched at least 50%). The fixing of the threshold aims to enhance the quality of the recommendations. For example, a video visualized only 20%, should not be recommended because the average view is very low which explains that the video could not interest the current user u_a so we should not recommend it to another similar user.

4.6.1.1 Evaluation Results

Now, we are going to transmit the evaluation results found after evaluating our developed algorithms. The figure 4.3 shows the precision scores of the Item Based and User Based algorithms basing on the k number of the nearest neighbors using Jaccard and Cosine similarities for the case of Binary View based evaluation.



(a) User Based

(b) Item Based

Figure 4.3: Precision scores for Memory Based algorithms using Binary View

As presented in the figure 4.3, the precision scores differ. The highest precision score for the Item Based approach is equal to 0.651 using Jaccard similarity with $k = 20$ nearest neighbors. The highest precision score of the User Based approach is equal to 0.644 using Jaccard similarity with $k = 20$ nearest neighbors. And, the Model Based SVD algorithm resulted an **average precision** equal to **0.634** basing on the Binary View.

4.6.1.2 Results Analysis

After testing the algorithms, we will display the best precision scores for the User Based, Item Based, and Model Based recommendation algorithms in the figure 4.4.

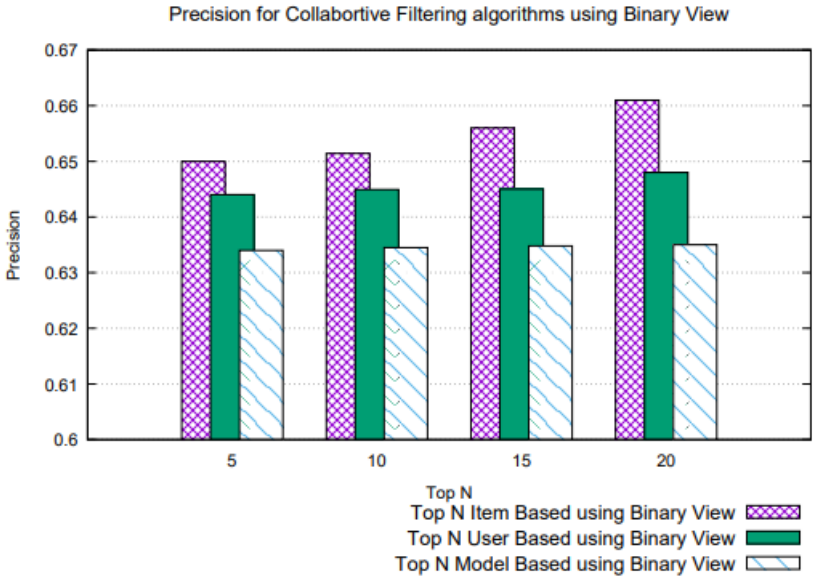


Figure 4.4: Precision scores for Collaborative Filtering algorithms using Binary View

Noting that the resulted precision scores are interesting, although, the **Item Based** algorithm shows the best **precision** equal to **0.651** using Jaccard similarity, $k = 20$, $threshold = 0.5$, and $N = 10$ basing on the Binary View.

4.6.2 Relative Duration Based Evaluation

Similarly to the Binary View Based evaluation (section 4.6.1), we have split our dataset (Relative Duration Matrix) into train set and test set using the cross validation method. The evaluation of the Memory Based algorithms was processed by varying the number of k nearest

neighbors and the similarity coefficient (Jaccard and Cosine).

4.6.2.1 Evaluation Results

The evaluation results of the Memory Based algorithms displayed using the RMSE scores resulted by varying the k number of nearest neighbors and by applying Jaccard and Cosine similarities are illustrated in the figure 4.5.

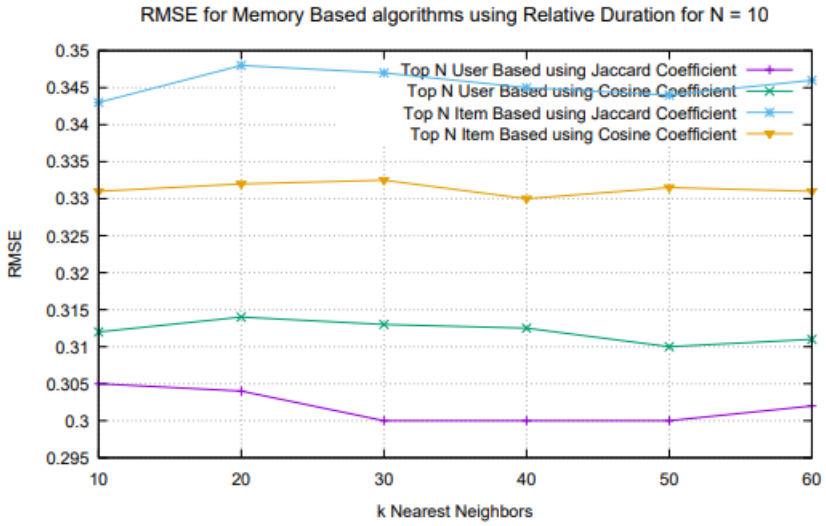


Figure 4.5: RMSE for Memory Based algorithms using Relative Duration

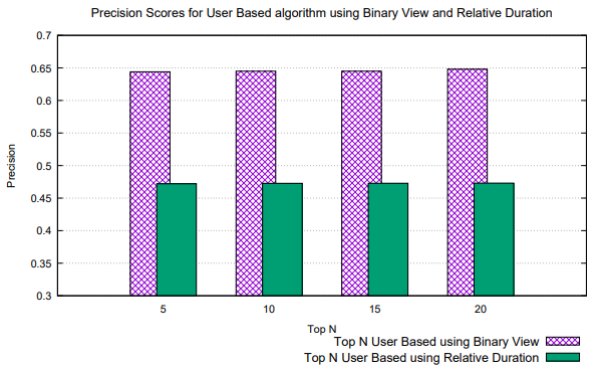
As presented in the figure 4.5, the RMSE scores differ for every k number of nearest neighbors and for the used similarity coefficient. The lowest RMSE for the User Based algorithm is equal to 0.3 using Jaccard similarity with $k \in \{30, 40, 50\}$. The lowest RMSE for the Item Based algorithm is equal to 0.33 using Jaccard similarity for $k = 40$. The Model Based SVD algorithm gives interesting results by an **average RMSE** equal to **0.15**.

4.6.2.2 Results Analysis

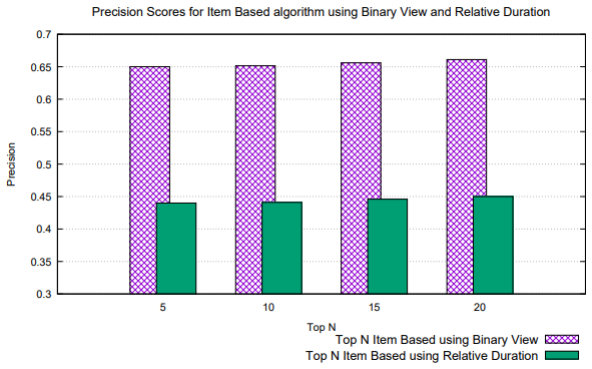
Obviously, the RMSE scores reflect interesting results for all the tested Collaborative Filtering algorithms. Although, the best recommendation algorithm basing on the Relative Duration is clearly the **SVD Model Based** algorithm that gives the lowest **RMSE** score equal to **0.15**.

4.6.3 Binary View and Relative Duration based evaluation

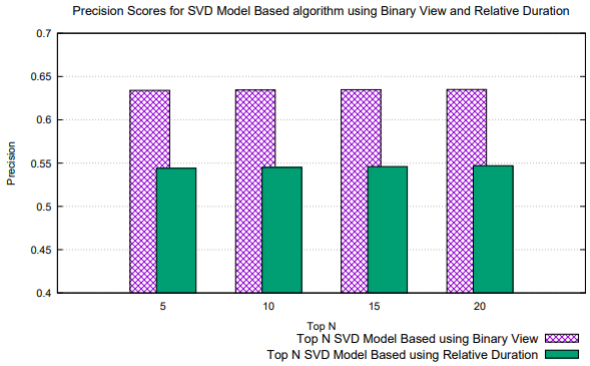
We have evaluated the personalized algorithms basing on the Binary View and Relative Duration highlighting the best algorithms resulting the highest precision scores for the case of Binary View and the lowest RMSE scores for the case of Relative Duration. Hence, we are going to plot, for every approach, the best precision scores found for the case of Binary View and Relative Duration as shown in the figure 4.6.



(a) User Based



(b) Item Based



(c) SVD Model Based

Figure 4.6: Precision for CF algorithms Based on Binary View and Relative Duration

From the presented figure 4.6, we can deduce some interpretations:

- Varying the number of Top N recommendations influence the quality of the recommender. So, if the number of Top N increases, the precision scores increase.
- For all the tested Collaborative Filtering algorithms (Item Based, User Based, and SVD Model Based), the found results emphasize that the use of Binary View gives higher precision than the use of the Relative Duration. So, we can conclude that we do not need to

calculate the relative durations that require more computational time and provide lower precision scores than the binary view durations.

- The best Collaborative Filtering algorithm for our case of study basing on the Binary View and Relative Duration is clearly the Item Based algorithm using the Binary View and presenting a precision score up to 0.651 for $k = 20$ nearest neighbors.

4.7 Discussion

The evaluation of the non personalized algorithms highlighted that the Most Recent algorithm is the best non personalized algorithm for our case of study showing the best Hit Rate Precision which is 0.485 for $N = 20$. And, in order to extract the best personalized recommendation algorithm, we are going to present the best recommendation algorithm found using the Binary View and the best recommendation algorithm found using the Relative Duration as shown in the figure 4.7.

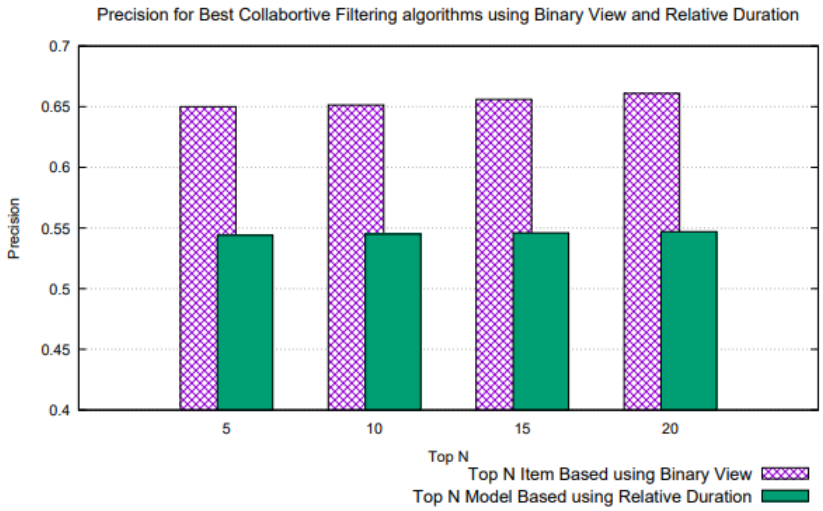


Figure 4.7: Precision for the best CF algorithms using the Binary View and the Relative Duration

The figure 4.7 shows that the best personalized recommendation algorithm used for our case of study is clearly the Item Based algorithm applying Jaccard similarity, $k = 20$ nearest neighbors, $threshold = 0.5$ (average view threshold), and $N = 10$ resulting a precision score equal to 0.651 using the Binary View Matrix.

Conclusion

Our proposed recommender system consists of a set of recommendation algorithms that are tested and evaluated using our provided usage data. The evaluation process is carried out for the case of non personalized and personalized recommendation algorithms to emphasize the quality of the recommender and highlight the best recommendation algorithm for both cases.

The chapter presented the used dataset in the section 4.1. Then, it introduced the used technologies and tool for realizing our work in the section 4.2. After, it defined the non personalized recommendation algorithms in the section 4.3 with their evaluations in the section 4.4. And, it presented the personalized recommendation algorithms in the section 4.5 with their evaluations in the section 4.6. Ending by discussing the results found in the section 4.7.

General Conclusion

The main goal of this project is to develop a recommendation engine for SmartVideo platform that streams Live and Video on Demand content of diverse local channels of the Grand Est Region of France. That's why, we have studied the state of the art of recommender systems to know how can these recommenders be utilized in the Smart Video field. Besides, we have focused on the Data Collection issue of recommender systems in order to find a solution to alleviate this highly time consuming task.

Focusing on the data collection task of recommender systems, we have explored the main issues of using the traditional Extract, Transform, and Load (ETL) process that needs a lot of time and expertise to be performed. Hence, we have proposed our approach used in order to facilitate the data presentation and collection tasks. Our approach consists of adopting the xAPI standard used to present, trace, store, gather, manage, and analyze data easily in a common and well structured format.

More precisely, we have adjusted the xAPI standard to fit our case of study so to be able to represent all the possible kinds of interactions a user perform to a video. As a result, we have introduced our xAPI video Recipe that consists of presenting the usage data about interactions in a formal and consistent structure. Moreover, we have presented the SmartVideo workflow while gathering data and performing recommendations detailing the possible communication methods between xAPI and SmartVideo platform.

Once we finished dealing with the Data Collection task, we have surveyed the recommender systems in depth to perform recommendations for users while their navigations. Therefore, we have defined a bunch of non personalized and personalized recommendation algorithms that we have developed and tested using our provided usage data. Certainly, the eval-

uation results allow us to get some insights about the type of our usage data at first, and about our developed recommender system then.

Thus, we have discussed the evaluation results found in order to highlight the best non personalized and personalized recommendation algorithms for our case. We could ensure that the results found emphasize the quality of our recommender system applying our usage data. Although, we have dealt only with the visualization activity while testing the recommendation algorithms, so we can propose, for further tasks, the use of other kinds of interactions or evaluations to enhance our recommender.

Notably, we propose some perspectives in order to enhance our work:

Enrich the Usage Data collected: Noting that we have utilized only the visualization task to apply recommendations and evaluate results. We can absolutely enrich our usage data by collecting other kind of usage data. These usage data can be presented by ratings, sharings, tags and comments, favorites, added to playlist, deleted from playlist, like/dislike, etc. All these data can be easily and simply presented using our proposed xAPI video recipe and they can ameliorate the recommendation results.

Apply more recommendation algorithms: Another purpose is to enrich our recommender system by developing other recommendation algorithms considering other features such as Content Based algorithm and hybridization method that combines collaborative filtering approach with content based approach. Also, a Diversity Based [Adomavicius and Kwon, 2011] recommendation algorithm can be performed to recommend similar content basing on different media types (news, articles, actualities, reportages, etc).

Bibliography

[Adomavicius and Kwon, 2011] Adomavicius, G. and Kwon, Y. (2011). Improving aggregate recommendation diversity using ranking-based techniques. IEEE Transactions on Knowledge and Data Engineering, 24(5):896–911.

[Adomavicius and Tuzhilin, 2005a] Adomavicius, G. and Tuzhilin, A. (2005a). Personalization technologies: a process-oriented perspective. Communications of the ACM, 48(10):83–90.

[Adomavicius and Tuzhilin, 2005b] Adomavicius, G. and Tuzhilin, A. (2005b). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE transactions on knowledge and data engineering, 17(6):734–749.

[Aggarwal et al., 2016] Aggarwal, C. C. et al. (2016). Recommender systems, volume 1. Springer.

[Aharon et al., 2015] Aharon, M., Hillel, E., Kagian, A., Lempel, R., Makabee, H., and Nissim, R. (2015). Watch-it-next: a contextual tv recommendation system. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pages 180–195. Springer.

[Aïmeur et al., 2008] Aïmeur, E., Brassard, G., Fernandez, J. M., and Onana, F. S. M. (2008). A lambic: a privacy-preserving recommender system for electronic commerce. International Journal of Information Security, 7(5):307–334.

[Ames and Naaman, 2007] Ames, M. and Naaman, M. (2007). Why we tag: motivations for annotation in mobile and online media. In Proceedings of the SIGCHI conference on Human factors in computing systems, pages 971–980.

[Anand and Mobasher, 2003] Anand, S. S. and Mobasher, B. (2003). Intelligent techniques for web personalization. In IJCAI Workshop on Intelligent Techniques for Web Personalization, pages 1–36. Springer.

[Anderson and Hiralall, 2009] Anderson, C. and Hiralall, M. (2009). Recommender systems for e-shops. Business Mathematics and Informatics paper.

[Ar and Bostanci, 2016] Ar, Y. and Bostanci, E. (2016). A genetic algorithm solution to the collaborative filtering problem. Expert Systems with Applications, 61:122–128.

[Bakharia et al., 2016] Bakharia, A., Kitto, K., Pardo, A., Gašević, D., and Dawson, S. (2016). Recipe for success: lessons learnt from using xapi within the connected learning analytics toolkit. In Proceedings of the sixth international conference on learning analytics & knowledge, pages 378–382.

[Bakshi et al., 2014] Bakshi, S., Jagadev, A. K., Dehuri, S., and Wang, G.-N. (2014). Enhancing scalability and accuracy of recommendation systems using unsupervised learning and particle swarm optimization. Applied Soft Computing, 15:21–29.

[Baroni et al., 2010] Baroni, M., Murphy, B., Barbu, E., and Poesio, M. (2010). Strudel: A corpus-based semantic model based on properties and types. Cognitive science, 34(2):222–254.

[Basu et al., 2016] Basu, S., Yu, Y., Singh, V. K., and Zimmermann, R. (2016). Videopedia: Lecture video recommendation for educational blogs using topic modeling. In International Conference on Multimedia Modeling, pages 238–250. Springer.

[Ben-Arieh and Chen, 2006] Ben-Arieh, D. and Chen, Z. (2006). Linguistic-labels aggregation and consensus measure for autocratic decision making using group recommendations. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, 36(3):558–568.

[Bennett et al., 2007] Bennett, J., Lanning, S., et al. (2007). The netflix prize. In Proceedings of KDD cup and workshop, volume 2007, page 35. New York.

[Berg et al., 2016] Berg, A., Scheffel, M., Drachsler, H., Ternier, S., and Specht, M. (2016). The dutch xapi experience. In Proceedings of the Sixth International Conference on Learning Analytics & Knowledge, pages 544–545.

[Berry et al., 1995] Berry, M. W., Dumais, S. T., and O'Brien, G. W. (1995). Using linear algebra for intelligent information retrieval. SIAM review, 37(4):573–595.

[Billsus et al., 1998] Billsus, D., Pazzani, M. J., et al. (1998). Learning collaborative information filters. In Icml, volume 98, pages 46–54.

[Blei et al., 2003] Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. Journal of machine Learning research, 3(Jan):993–1022.

[Brill et al., 2001] Brill, E., Lin, J., Banko, M., Dumais, S., Ng, A., et al. (2001). Data-intensive question answering. In TREC, volume 56, page 90.

[Brusilovsky and Millán, 2007] Brusilovsky, P. and Millán, E. (2007). User models for adaptive hypermedia and adaptive educational systems. In The adaptive web, pages 3–53. Springer.

[Burke, 2002a] Burke, R. (2002a). Hybrid recommender systems: Survey and experiments. User modeling and user-adapted interaction, 12(4):331–370.

[Burke, 2007] Burke, R. (2007). Hybrid web recommender systems. In The adaptive web, pages 377–408. Springer.

[Burke, 2002b] Burke, R. R. (2002b). Technology and the customer interface: what consumers want in the physical and virtual store. Journal of the academy of Marketing Science, 30(4):411–432.

[Castro et al., 2017] Castro, J., Lu, J., Zhang, G., Dong, Y., and Martínez, L. (2017). Opinion dynamics-based group recommender systems. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 48(12):2394–2406.

[Chang et al., 2013] Chang, N., Irvan, M., and Terano, T. (2013). A tv program recommender framework. Procedia Computer Science, 22:561–570.

[Chen et al., 2015] Chen, L., Zhou, Y., and Chiu, D. M. (2015). Smart streaming for online video services. IEEE transactions on multimedia, 17(4):485–497.

[Chen et al., 2011] Chen, Y., Wu, C., Xie, M., and Guo, X. (2011). Solving the sparsity problem in recommender systems using association retrieval. Journal of computers, 6(9):1896–1902.

[Chidlovskii et al., 2001] Chidlovskii, B., Glance, N. S., and Grasso, A. (2001). System and method for collaborative ranking of search results employing user and group profiles derived from document collection content analysis. US Patent 6,327,590.

[Costello and Osborne, 2005] Costello, A. B. and Osborne, J. (2005). Best practices in exploratory factor analysis: Four recommendations for getting the most from your analysis. Practical assessment, research, and evaluation, 10(1):7.

[Covington et al., 2016] Covington, P., Adams, J., and Sargin, E. (2016). Deep neural networks for youtube recommendations. In Proceedings of the 10th ACM conference on recommender systems, pages 191–198.

[Cremonesi et al., 2010] Cremonesi, P., Koren, Y., and Turrin, R. (2010). Performance of recommender algorithms on top-n recommendation tasks. In Proceedings of the fourth ACM conference on Recommender systems, pages 39–46.

[Davidson et al., 2010] Davidson, J., Liebald, B., Liu, J., Nandy, P., Van Vleet, T., Gargi, U., Gupta, S., He, Y., Lambert, M., Livingston, B., et al. (2010). The youtube video recommendation system. In Proceedings of the fourth ACM conference on Recommender systems, pages 293–296.

[De Gemmis et al., 2015] De Gemmis, M., Lops, P., Musto, C., Narducci, F., and Semeraro, G. (2015). Semantics-aware content-based recommender systems. In Recommender systems handbook, pages 119–159. Springer.

[De Lathauwer et al., 2000] De Lathauwer, L., De Moor, B., and Vandewalle, J. (2000). A multilinear singular value decomposition. SIAM journal on Matrix Analysis and Applications, 21(4):1253–1278.

[De Pessemier et al., 2016] De Pessemier, T., Verlee, D., and Martens, L. (2016). Enhancing recommender systems for tv by face recognition. In 12th international conference on web information systems and technologies (WEBIST 2016), volume 2, pages 243–250.

[Deerwester et al., 1990] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. Journal of the American society for information science, 41(6):391–407.

[Deldjoo et al., 2018] Deldjoo, Y., Constantin, M. G., Eghbal-Zadeh, H., Ionescu, B., Schedl, M., and Cremonesi, P. (2018). Audio-visual encoding of multimedia content for enhancing movie recommendations. In Proceedings of the 12th ACM Conference on Recommender Systems, pages 455–459.

[Deldjoo et al., 2016] Deldjoo, Y., Elahi, M., Cremonesi, P., Garzotto, F., Piazzolla, P., and Quadrana, M. (2016). Content-based video recommendation system based on stylistic visual features. Journal on Data Semantics, 5(2):99–113.

[Deldjoo et al., 2019] Deldjoo, Y., Schedl, M., and Elahi, M. (2019). Movie genome recommender: A novel recommender system based on multimedia content. In 2019 International Conference on Content-Based Multimedia Indexing (CBMI), pages 1–4. IEEE.

[Delgado and Davidson, 2002] Delgado, J. and Davidson, R. (2002). Knowledge bases and user profiling in travel and hospitality recommender systems. In ENTER, pages 1–16. Citeseer.

[Demiriz, 2004] Demiriz, A. (2004). Enhancing product recommender systems on sparse binary data. Data Mining and Knowledge Discovery, 9(2):147–170.

[Deshpande and Karypis, 2004] Deshpande, M. and Karypis, G. (2004). Item-based top-n recommendation algorithms. ACM Transactions on Information Systems (TOIS), 22(1):143–177.

[Desrosiers and Karypis, 2011] Desrosiers, C. and Karypis, G. (2011). A comprehensive survey of neighborhood-based recommendation methods. In Recommender systems handbook, pages 107–144. Springer.

[Do et al., 2010] Do, M.-P. T., Nguyen, D., and Nguyen, L. (2010). Model-based approach for collaborative filtering. In 6th International Conference on Information Technology for Education.

[Ekstrand et al., 2011] Ekstrand, M. D., Riedl, J. T., and Konstan, J. A. (2011). Collaborative filtering recommender systems. Now Publishers Inc.

[El Akkaoui et al., 2011] El Akkaoui, Z., Zimànyi, E., Mazón, J.-N., and Trujillo, J. (2011). A model-driven framework for etl process development. In Proceedings of the ACM 14th international workshop on Data Warehousing and OLAP, pages 45–52.

[Evangelopoulos et al., 2012] Evangelopoulos, N., Zhang, X., and Prybutok, V. R. (2012). Latent semantic analysis: five methodological recommendations. European Journal of Information Systems, 21(1):70–86.

[Fournier-Viger et al., 2017] Fournier-Viger, P., Lin, J. C.-W., Kiran, R. U., Koh, Y. S., and Thomas, R. (2017). A survey of sequential pattern mining. Data Science and Pattern Recognition, 1(1):54–77.

[Frazier et al., 1998] Frazier, C. J., Rollins, Y. D., Breese, C. R., Leonard, S., Freedman, R., and Dunwiddie, T. V. (1998). Acetylcholine activates an α-bungarotoxin-sensitive nicotinic current in rat hippocampal interneurons, but not pyramidal cells. Journal of Neuroscience, 18(4):1187–1195.

[Ganguly et al., 2015] Ganguly, D., Roy, D., Mitra, M., and Jones, G. J. (2015). Word embedding based generalized language model for information retrieval. In Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval, pages 795–798.

[Ghazanfar and Prugel-Bennett, 2011] Ghazanfar, M. and Prugel-Bennett, A. (2011). Fulfilling the needs of gray-sheep users in recommender systems, a clustering solution.

[Ghazanfar and Prugel-Bennett, 2010] Ghazanfar, M. A. and Prugel-Bennett, A. (2010). A scalable, accurate hybrid recommender system. In 2010 Third International Conference on Knowledge Discovery and Data Mining, pages 94–98. IEEE.

[Ghazanfar and Prügel-Bennett, 2014] Ghazanfar, M. A. and Prügel-Bennett, A. (2014). Leveraging clustering approaches to solve the gray-sheep users problem in recommender systems. Expert Systems with Applications, 41(7):3261–3275.

[Gilotte et al., 2018] Gilotte, A., Calauzènes, C., Nedelec, T., Abraham, A., and Dollé, S. (2018). Offline a/b testing for recommender systems. In Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, pages 198–206.

[Gomez-Uribe and Hunt, 2015] Gomez-Uribe, C. A. and Hunt, N. (2015). The netflix recommender system: Algorithms, business value, and innovation. ACM Transactions on Management Information Systems (TMIS), 6(4):1–19.

[Grčar et al., 2006] Grčar, M., Fortuna, B., Mladenič, D., and Grobelnik, M. (2006). knn versus svm in the collaborative filtering framework. In Data Science and Classification, pages 251–260. Springer.

[Gretzel and Fesenmaier, 2006] Gretzel, U. and Fesenmaier, D. R. (2006). Persuasion in recommender systems. International Journal of Electronic Commerce, 11(2):81–100.

[Hanani et al., 2001] Hanani, U., Shapira, B., and Shoval, P. (2001). Information filtering: Overview of issues, research and systems. User modeling and user-adapted interaction, 11(3):203–259.

[Hartsell and Yuen, 2006] Hartsell, T. and Yuen, S. C.-Y. (2006). Video streaming in online learning. AACE Journal, 14(1):31–43.

[Hegselmann et al., 2002] Hegselmann, R., Krause, U., et al. (2002). Opinion dynamics and bounded confidence models, analysis, and simulation. Journal of artificial societies and social simulation, 5(3).

[Herlocker et al., 2004] Herlocker, J. L., Konstan, J. A., Terveen, L. G., and Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems (TOIS), 22(1):5–53.

[Hicken et al., 2005] Hicken, W., Holm, F., Clune, J., and Campbell, M. (2005). Music recommendation system and method. US Patent App. 10/917,865.

[Ho et al., 2014] Ho, C.-Y., Meng, I.-H., and Tsai, C.-H. (2014). Video recommendation system and method thereof. US Patent 8,804,999.

[Hongliang and Xiaona, 2015] Hongliang, C. and Xiaona, Q. (2015). The video recommendation system based on dbn. In 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing, pages 1016–1021. IEEE.

[Huang et al., 2004] Huang, Z., Chen, H., and Zeng, D. (2004). Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. ACM Transactions on Information Systems (TOIS), 22(1):116–142.

[Iaquinta et al., 2008] Iaquinta, L., De Gemmis, M., Lops, P., Semeraro, G., Filannino, M., and Molino, P. (2008). Introducing serendipity in a content-based recommender system. In 2008 Eighth International Conference on Hybrid Intelligent Systems, pages 168–173. IEEE.

[Isinkaye et al., 2015] Isinkaye, F., Folajimi, Y., and Ojokoh, B. (2015). Recommendation systems: Principles, methods and evaluation. Egyptian Informatics Journal, 16(3):261–273.

[Iyer et al., 2015] Iyer, N., Dcunha, A., Desai, A., and Jain, K. (2015). Survey on online recommendation using web usage mining. International Journal of Computer Science and Information Technologies, 6(2):1465–7.

[Jannach et al., 2010] Jannach, D., Zanker, M., Felfernig, A., and Friedrich, G. (2010). Recommender systems: an introduction. Cambridge University Press.

[Jeckmans et al., 2013] Jeckmans, A. J., Beye, M., Erkin, Z., Hartel, P., Lagendijk, R. L., and Tang, Q. (2013). Privacy in recommender systems. In Social media retrieval, pages 263–281. Springer.

[Jenatton et al., 2012] Jenatton, R., Roux, N., Bordes, A., and Obozinski, G. R. (2012). A latent factor model for highly multi-relational data. Advances in neural information processing systems, 25:3167–3175.

[Jenner, 2017] Jenner, M. (2017). Binge-watching: Video-on-demand, quality tv and mainstreaming fandom. International Journal of Cultural Studies, 20(3):304–320.

[Jia et al., 2015] Jia, X., Wang, A., Li, X., Xun, G., Xu, W., and Zhang, A. (2015). Multimodal learning for video recommendation based on mobile application usage. In 2015 IEEE International Conference on Big Data (Big Data), pages 837–842. IEEE.

[Jones and Norrander, 1996] Jones, B. S. and Norrander, B. (1996). The reliability of aggregated public opinion measures. American Journal of Political Science, pages 295–309.

[Juluri et al., 2015] Juluri, P., Tamarapalli, V., and Medhi, D. (2015). Measurement of quality of experience of video-on-demand services: A survey. IEEE Communications Surveys & Tutorials, 18(1):401–418.

[Kevan and Ryan, 2016] Kevan, J. M. and Ryan, P. R. (2016). Experience api: Flexible, decentralized and activity-centric data collection. Technology, knowledge and learning, 21(1):143–149.

[Khatwani and Chandak, 2016] Khatwani, S. and Chandak, M. (2016). Building personalized and non personalized recommendation systems. In 2016 International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT), pages 623–628. IEEE.

[Khusro et al., 2016] Khusro, S., Ali, Z., and Ullah, I. (2016). Recommender systems: issues, challenges, and research opportunities. In Information Science and Applications (ICISA) 2016, pages 1179–1189. Springer.

[Konstan, 2008] Konstan, J. A. (2008). Introduction to recommender systems. In Proceedings of the 2008 ACM SIGMOD international conference on Management of data, pages 1373–1374.

[Koren et al., 2009] Koren, Y., Bell, R., and Volinsky, C. (2009). Matrix factorization techniques for recommender systems. Computer, 42(8):30–37.

[Krestel et al., 2009] Krestel, R., Fankhauser, P., and Nejdl, W. (2009). Latent dirichlet allocation for tag recommendation. In Proceedings of the third ACM conference on Recommender systems, pages 61–68.

[Krulwich, 1997] Krulwich, B. (1997). Lifestyle finder: Intelligent user profiling using large-scale demographic data. AI magazine, 18(2):37–37.

[Labba et al., 2019] Labba, C., Roussanaly, A., and Boyer, A. (2019). Towards an automated framework for benchmarking learning record stores: Performance requirements and scalability. In 3rd Annual Learning & Student Analytics Conference (LSAC 2019).

[Labba et al., 2020] Labba, C., Roussanaly, A., and Boyer, A. (2020). An operational framework for evaluating the performance of learning record stores. In European Conference on Technology Enhanced Learning, pages 45–59. Springer.

[LANGab et al., 2008] LANGab, M., Bürkle, T., Laumann, S., and Prokosch, H.-U. (2008). Process mining for clinical workflows: challenges and current limitations. In EHealth beyond the horizon: Get it there: Proceedings of MIE2008 the XXIst international congress of the european federation for medical informatics, page 229.

[Lian et al., 2017] Lian, T., Li, Z., Chen, Z., and Ma, J. (2017). The impact of profile coherence on recommendation performance for shared accounts on smart tvs. In China Conference on Information Retrieval, pages 30–41. Springer.

[Lika et al., 2014] Lika, B., Kolomvatsos, K., and Hadjiefthymiades, S. (2014). Facing the cold start problem in recommender systems. Expert Systems with Applications, 41(4):2065–2073.

[Lim, 2015] Lim, K. C. (2015). Case studies of xapi applications to e-learning. In The Twelfth International Conference on eLearning for Knowledge-Based Society, pages 3–1.

[Lim, 2016] Lim, K. C. (2016). Using xapi and learning analytics in education. In Elearning Forum Asia, pages 13–15.

[Lim, 2018] Lim, K. C. (2018). Using the xapi to track learning. In Innovations in Open and Flexible Education, pages 233–242. Springer.

[Linden et al., 2003] Linden, G., Smith, B., and York, J. (2003). Amazon. com recommendations: Item-to-item collaborative filtering. IEEE Internet computing, 7(1):76–80.

[Liu et al., 2016] Liu, P., Ma, J., Wang, Y., Ma, L., and Huang, S. (2016). A context-aware method for top-k recommendation in smart tv. In Asia-Pacific Web Conference, pages 150–161. Springer.

[Lops et al., 2011] Lops, P., De Gemmis, M., and Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. In Recommender systems handbook, pages 73–105. Springer.

[Lousame and Sánchez, 2009] Lousame, F. P. and Sánchez, E. (2009). A taxonomy of collaborative-based recommender systems. In Web Personalization in Intelligent Environments, pages 81–117. Springer.

[Lu et al., 2015] Lu, J., Wu, D., Mao, M., Wang, W., and Zhang, G. (2015). Recommender system application developments: a survey. Decision Support Systems, 74:12–32.

[Manso-Vazquez et al., 2015] Manso-Vazquez, M., Caeiro-Rodriguez, M., and Llamas-Nistal, M. (2015). xapi-srl: Uses of an application profile for self-regulated learning based on the analysis of learning strategies. In 2015 IEEE Frontiers in Education Conference (FIE), pages 1–8. IEEE.

[Masthoff, 2015] Masthoff, J. (2015). Group recommender systems: aggregation, satisfaction and group attributes. In recommender systems handbook, pages 743–776. Springer.

[Mei et al., 2009] Mei, T., Hua, X.-S., Yang, B., Yang, L., and Li, S. (2009). Automatic video recommendation. US Patent App. 11/771,219.

[Mell et al., 2011] Mell, P., Grance, T., et al. (2011). The nist definition of cloud computing.

[Melville-Jones, 2002] Melville-Jones, J. (2002). Venice and Thessalonica 1423-1430: The Venetian Documents. Unipress Padova.

[Micaelian et al., 2004] Micaelian, F. V., Sawey, R., Scoffone, E. M., and Criswell, D. B. (2004). Weighted preference data search system and method. US Patent 6,714,929.

[Mukhopadhyay et al., 2008] Mukhopadhyay, D., Dutta, R., Kundu, A., and Dattagupta, R. (2008). A product recommendation system using vector space model and association rule. In 2008 International Conference on Information Technology, pages 279–282. IEEE.

[Musto et al., 2016] Musto, C., Semeraro, G., de Gemmis, M., and Lops, P. (2016). Learning word embeddings from wikipedia for content-based recommender systems. In European Conference on Information Retrieval, pages 729–734. Springer.

[Niwattanakul et al., 2013] Niwattanakul, S., Singthongchai, J., Naenudorn, E., and Wanapu, S. (2013). Using of jaccard coefficient for keywords similarity. In Proceedings of the international multiconference of engineers and computer scientists, volume 1, pages 380–384.

[Nouira et al., 2018] Nouira, A., Cheniti-Belcadhi, L., and Braham, R. (2018). An enhanced xapi data model supporting assessment analytics. Procedia Computer Science, 126:566–575.

[Otebolaku and Andrade, 2017] Otebolaku, A. M. and Andrade, M. T. (2017). Context-aware personalization using neighborhood-based context similarity. Wireless Personal Communications, 94(3):1595–1618.

[Pan, 2017] Pan, S.-W. (2017). Smart tv system and input operation method. US Patent 9,729,811.

[Papagelis et al., 2005] Papagelis, M., Plexousakis, D., and Kutsuras, T. (2005). Alleviating the sparsity problem of collaborative filtering using trust inferences. In International conference on trust management, pages 224–239. Springer.

[Pazzani and Billsus, 2007] Pazzani, M. J. and Billsus, D. (2007). Content-based recommendation systems. In The adaptive web, pages 325–341. Springer.

[Peis et al., 2008] Peis, E., del Castillo, J. M., and Delgado-López, J. A. (2008). Semantic recommender systems. analysis of the state of the topic. Hipertext. net, 6(2008):1–5.

[Pereira and Hruschka, 2015] Pereira, A. L. V. and Hruschka, E. R. (2015). Simultaneous co-clustering and learning to address the cold start problem in recommender systems. Knowledge-Based Systems, 82:11–19.

[Poriya et al., 2014] Poriya, A., Bhagat, T., Patel, N., and Sharma, R. (2014). Non-personalized recommender systems and user-based collaborative recommender systems. Int. J. Appl. Inf. Syst, 6(9):22–27.

[Prakash and Rangdale, 2017] Prakash, G. H. and Rangdale, S. (2017). Etl data conversion: Extraction, transformation and loading data conversion. International Journal Of Engineering And Computer Science, 6(10):22545–22550.

[Pyo et al., 2013] Pyo, S., Kim, E., and Kim, M. (2013). Automatic and personalized recommendation of tv program contents using sequential pattern mining for smart tv user interaction. Multimedia systems, 19(6):527–542.

[Resnick et al., 1994] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. (1994). Grouplens: an open architecture for collaborative filtering of netnews. In Proceedings of the 1994 ACM conference on Computer supported cooperative work, pages 175–186.

[Safoury and Salah, 2013] Safoury, L. and Salah, A. (2013). Exploiting user demographic attributes for solving cold-start problem in recommender system. Lecture Notes on Software Engineering, 1(3):303–307.

[Salton et al., 1997] Salton, G., Singhal, A., Mitra, M., and Buckley, C. (1997). Automatic text structuring and summarization. Information processing & management, 33(2):193–207.

[Sarwar et al., 2000a] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. (2000a). Analysis of recommendation algorithms for e-commerce. In Proceedings of the 2nd ACM conference on Electronic commerce, pages 158–167.

[Sarwar et al., 2000b] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. (2000b). Application of dimensionality reduction in recommender system-a case study. Technical report, Minnesota Univ Minneapolis Dept of Computer Science.

[Sarwar et al., 2001] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th international conference on World Wide Web, pages 285–295.

[Sarwar et al., 2002] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. (2002). Incremental singular value decomposition algorithms for highly scalable recommender systems. In Fifth international conference on computer and information science, volume 1, pages 27–8. Citeseer.

[Sarwar, 2001] Sarwar, B. M. (2001). Sparsity, scalability, and distribution in recommender systems.

[Schafer et al., 2007] Schafer, J. B., Frankowski, D., Herlocker, J., and Sen, S. (2007). Collaborative filtering recommender systems. In The adaptive web, pages 291–324. Springer.

[Schafer et al., 1999] Schafer, J. B., Konstan, J., and Riedl, J. (1999). Recommender systems in e-commerce. In Proceedings of the 1st ACM conference on Electronic commerce, pages 158–166.

[Schein et al., 2002] Schein, A. I., Popescul, A., Ungar, L. H., and Pennock, D. M. (2002). Methods and metrics for cold-start recommendations. In Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval, pages 253–260.

[Seo et al., 2020] Seo, Y.-D., Lee, E., and Kim, Y.-G. (2020). Video on demand recommender system for internet protocol television service based on explicit information fusion. Expert Systems with Applications, 143:113045.

[Serrano-Laguna et al., 2017] Serrano-Laguna, Á., Martínez-Ortiz, I., Haag, J., Regan, D., Johnson, A., and Fernández-Manjón, B. (2017). Applying standards to systematize learning analytics in serious games. Computer Standards & Interfaces, 50:116–123.

[Shani and Gunawardana, 2011] Shani, G. and Gunawardana, A. (2011). Evaluating recommendation systems. In Recommender systems handbook, pages 257–297. Springer.

[Shani et al., 2005] Shani, G., Heckerman, D., and Brafman, R. I. (2005). An mdp-based recommender system. Journal of Machine Learning Research, 6(Sep):1265–1295.

[Sharma and Gera, 2013] Sharma, L. and Gera, A. (2013). A survey of recommendation system: Research challenges. International Journal of Engineering Trends and Technology (IJETT), 4(5):1989–1992.

[Smith and Linden, 2017] Smith, B. and Linden, G. (2017). Two decades of recommender systems at amazon. com. Ieee internet computing, 21(3):12–18.

[Smith et al., 2013] Smith, T., Obrist, M., and Wright, P. (2013). Live-streaming changes the (video) game. In Proceedings of the 11th european conference on Interactive TV and video, pages 131–138.

[Spiliopoulou et al., 2003] Spiliopoulou, M., Mobasher, B., Berendt, B., and Nakagawa, M. (2003). A framework for the evaluation of session reconstruction heuristics in web-usage analysis. Informs journal on computing, 15(2):171–190.

[Su and Khoshgoftaar, 2009] Su, X. and Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. Advances in artificial intelligence, 2009.

[Thada and Jaglan, 2013] Thada, V. and Jaglan, V. (2013). Comparison of jaccard, dice, cosine similarity coefficient to find best fitness value for web retrieved documents using genetic algorithm. International Journal of Innovations in Engineering and Technology, 2(4):202–205.

[Thorat et al., 2015] Thorat, P. B., Goudar, R., and Barve, S. (2015). Survey on collaborative filtering, content-based filtering and hybrid recommendation system. International Journal of Computer Applications, 110(4):31–36.

[Ticha, 2015] Ticha, S. B. (2015). Recommandation personnalisée hybride. PhD thesis.

[Ungar and Foster, 1998] Ungar, L. H. and Foster, D. P. (1998). Clustering methods for collaborative filtering. In AAAI workshop on recommendation systems, volume 1, pages 114–129. Menlo Park, CA.

[Van den Oord et al., 2013] Van den Oord, A., Dieleman, S., and Schrauwen, B. (2013). Deep content-based music recommendation. In Advances in neural information processing systems, pages 2643–2651.

[Verma et al., 2015] Verma, J. P., Patel, B., and Patel, A. (2015). Big data analysis: recommendation system with hadoop framework. In 2015 IEEE International Conference on Computational Intelligence & Communication Technology, pages 92–97. IEEE.

[Vijayakumar et al., 2019] Vijayakumar, V., Vairavasundaram, S., Logesh, R., and Sivapathi, A. (2019). Effective knowledge based recommender system for tailored multiple point of interest recommendation. International Journal of Web Portals (IJWP), 11(1):1–18.

[Wall et al., 2003] Wall, M. E., Rechtsteiner, A., and Rocha, L. M. (2003). Singular value decomposition and principal component analysis. In A practical approach to microarray data analysis, pages 91–109. Springer.

[Woerndl et al., 2007] Woerndl, W., Schueller, C., and Wojtech, R. (2007). A hybrid recommender system for context-aware recommendations of mobile applications. In 2007 IEEE 23rd International Conference on Data Engineering Workshop, pages 871–878. IEEE.

[Xiao et al., 2018] Xiao, J., Wang, M., Jiang, B., and Li, J. (2018). A personalized recommendation system with combinational algorithm for online learning. Journal of Ambient Intelligence and Humanized Computing, 9(3):667–677.

[Xiaojian et al., 2014] Xiaojian, Z., Miao, W., and Xiaofeng, M. (2014). An accurate method for mining top-k frequent pattern under differential privacy. Journal of Computer Research and Development, 51(1):104.

[Yildirim and Krishnamoorthy, 2008] Yildirim, H. and Krishnamoorthy, M. S. (2008). A random walk method for alleviating the sparsity problem in collaborative filtering. In Proceedings of the 2008 ACM conference on Recommender systems, pages 131–138.

[Yu et al., 2012] Yu, L., Zheng, J., Shen, W. C., Wu, B., Wang, B., Qian, L., and Zhang, B. R. (2012). Bc-pdm: data mining, social network analysis and text mining system based on cloud computing. In Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 1496–1499.

[Zenebe and Norcio, 2009] Zenebe, A. and Norcio, A. F. (2009). Representation, similarity measures and aggregation methods using fuzzy sets for content-based recommender systems. Fuzzy sets and systems, 160(1):76–94.

[Zhang et al., 2010] Zhang, Z.-K., Liu, C., Zhang, Y.-C., and Zhou, T. (2010). Solving the cold-start problem in recommender systems with social tags. EPL (Europhysics Letters), 92(2):28002.

[Zhou et al., 2016] Zhou, P., Zhou, Y., Wu, D., and Jin, H. (2016). Differentially private online learning for cloud-based video recommendation with multimedia big data in social networks. IEEE transactions on multimedia, 18(6):1217–1229.

[Zhou et al., 2010] Zhou, R., Khemmarat, S., and Gao, L. (2010). The impact of youtube recommendation system on video views. In Proceedings of the 10th ACM SIGCOMM conference on Internet measurement, pages 404–410.

[Zhou et al., 2019] Zhou, R., Xia, D., Wan, J., and Zhang, S. (2019). An intelligent video tag recommendation method for improving video popularity in mobile computing environment. IEEE Access, 8:6954–6967.

[Zhou et al., 2012] Zhou, X., Xu, Y., Li, Y., Josang, A., and Cox, C. (2012). The state-of-the-art in personalized recommender systems for social networking. Artificial Intelligence Review, 37(2):119–132.

[Zhou et al., 2008] Zhou, Y., Wilkinson, D., Schreiber, R., and Pan, R. (2008). Large-scale parallel collaborative filtering for the netflix prize. In International conference on algorithmic applications in management, pages 337–348. Springer.

Netography

[URL1] https://www.wired.co.uk/article/how-do-netflixs-algorithms-\
work-machine-learning-helps-to-predict-what-viewers-will-like, consulted in 2 December 2020.

[URL2] http://rejoiner.com/resources/amazon-recommendations-secret-\
selling-online/, consulted in 2 December 2020.

[URL3] http://outil2amenagement.cerema.fr/l-appel-a-manifestation-d-\
interet-ami-r295.html, consulted in 04 December 2020.

[URL4] https://www.kardham-digital.com/, consulted in 04 December 2020.

[URL5] https://www.viavosges.tv/, consulted in 04 December 2020.

[URL6] https://kiwi.loria.fr/, consulted in 04 December 2020.

[URL7] https://www.loria.fr/en/https://www.viavosges.tv/, consulted in 04 December 2020.

[URL8] https://encrypted-tbn0.gstatic.com/images?q=tbn%
3AANd9GcRdrWo0v8K_gJRIApm4iVMtJWokpPYoAP1NkQ&usqp=CAU, consulted in 15 September 2020.

[URL9] https://www.bigcommerce.com/blog/amazon-statistics/
#10-fascinating-amazon-statistics-sellers-need-to-know-in-2020, consulted in 2 December 2020.

[URL10] https://i.stack.imgur.com/4ncGL.png, consulted in 15 September 2020.

[URL11] https://panoply.io/data-warehouse-guide/3-ways-to-build-an-\
etl-process/, consulted in 15 November 2020.

[URL12] https://www.pcmag.com/picks/the-best-video-streaming-services?test_uuid=01jrZgWNXhmA3ocG7ZHXevj&test_variant=b, consulted in 25 September 2020.

[URL13] https://www.intellectsoft.net/blog/top-10-video-streaming-\platforms/, consulted in 17 September 2020.

[URL14] https://www.netflix.com/tn-en/, consulted in 05 September 2020.

[URL15] https://www.comparitech.com/tv-streaming/netflix-subscribers/, consulted in 11 September 2020.

[URL16] https://www.primevideo.com/?ref_=dvm_pds_amz_TN_kc_s_g|m_NX82yATpc_c415560809998, consulted in 29 September 2020.

[URL17] https://market.us/statistics/online-video-and-streaming-sites/amazon-prime-video/#:~:text=As%20of%20January%202020%2C%20there,over%20100%20million%20users%20globally, consulted in 19 September 2020.

[URL18] https://www.youtube.com/, consulted in 15 September 2020.

[URL19] https://hackernoon.com/youtubes-recommendation-engine-explained\-40j83183, consulted in 06 September 2020.

[URL20] https://www.canalplus.com/streaming/, consulted in 15 September 2020.

[URL21] https://spideo.com/en/portfolio/canal/, consulted in 12 September 2020.

[URL22] https://towardsdatascience.com/deep-dive-into-netflixs-\recommender-system-341806ae3b48, consulted in 2 December 2020.

[URL23] https://www.cnil.fr/fr/reglement-europeen-protection-donnees, consulted in 12 October 2020.

[URL24] https://github.com/adlnet/xAPI-Spec, consulted in 12 September 2020.

[URL25] https://github.com/adlnet/xAPI-Spec/blob/master/xAPI-About.md#partone, consulted in 08 September 2020.

[URL26] https://github.com/adlnet/xAPI-Spec/blob/master/xAPI-Data.md#parttwo, consulted in 17 September 2020.

[URL27] https://learningsolutionsmag.com/articles/10-best-practices-for-xapi-statements, consulted in 23 October 2020.

[URL28] https://learningsolutionsmag.com/sites/default/files/inline-images/xapi-statement-model-final.png, consulted in 21 October 2020.

[URL29] https://github.com/adlnet/xAPI-Spec/blob/master/xAPI-Communication.md#partthree, consulted in 28 November 2020.

[URL30] https://xapi.com/recipes/, consulted in 20 October 2020.

[URL31] https://xapi.com/recipes/?utm_source=google&utm_medium=natural_search, consulted in 25 October 2020.

[URL32] https://geojson.org/, consulted in 21 November 2020.

[URL33] https://www.learnupon.com/blog/what-is-an-lrs-learning-\record-store/, consulted in 30 October 2020.

[URL34] https://www.nextsoftwaresolutions.com/grassblade-lrs-\experience-api/, consulted in 30 October 2020.

[URL35] https://www.watershedlrs.com/, consulted in 08 October 2020.

[URL36] https://www.yetanalytics.com/, consulted in 15 October 2020.

[URL37] http://traxlrs.com/, consulted in 26 October 2020.

[URL38] https://learningpool.com/solutions/learning-record-store-\learning-locker/learning-locker-community-overview/, consulted in 21 October 2020.

[URL39] https://github.com/trax-project/trax-lrs, consulted in 06 November 2020.

[URL40] https://www.viavosges.tv/, consulted in 28 October 2020.

[URL41] https://www.alsace20.tv/, consulted in 17 October 2020.

[URL42] https://laravel.com/, consulted in 24 October 2020.

[URL43] https://flask.palletsprojects.com/en/1.1.x/, consulted in 1 December 2020.

[URL44] https://jupyter.org/, consulted in 4 December 2020.

[URL45] https://swagger.io/docs/specification/about/, consulted in 1 December 2020.

[URL46] http://www.gnuplot.info/, consulted in 4 December 2020.

[URL47] https://www.php.net/, consulted in 29 October 2020.

[URL48] https://www.python.org/, consulted in 04 December 2020.

[URL49] http://surpriselib.com/, consulted in 02 December 2020.

[URL50] https://pandas.pydata.org/, consulted in 03 December 2020.

[URL51] http://www.cs.carleton.edu/cs_comps/0607/recommend/recommender/images/svd1.png, consulted in 5 December 2020.