People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
Chadli Bendjedid University
Faculty of Science and Technology
IT department

الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي
جامعة الشاذلي بن جديد
كلية العلوم و التكنولوجيا
قسم الاعلام الالي

جامعة الشاذلي بن جديد
UNIVERSITE CHADLI BENDJEDID

Master Thesis in Computer Science

Presented by

AIOUAZ SAFIA

For the graduation of
MASTER
Filière : Informatique

Specialty: Intelligent Computer Systems

Theme

An Intelligent recommendation system based on context

Sustained on: July 2021

Quality	**Name and First Name**	**Degree**	**University**
President	Dr. ANGUEL F	MCB	Chadli Bendjedid El-Tarf
Supervisor	**Dr. MAATALLAH M**	**MCB**	**Chadli Bendjedid El-Tarf**
Examiner	Dr. CHAMAME C	MAB	Chadli Bendjedid El-Tarf

University Year: 2020/2021

Thanks

We thank Allah the Almighty, who has given us the strength and patience to do this work.

I thank the most sincere, our deep gratitude and I respects are addressed to our Coach

Dr. Majda Maatallah

For agreeing to supervise us, for the precious advice and guidance he provided during this Thesis.

Without his help, our work would not have seen the light.

We sincerely thank the members of the jury who did us the honor of accepting to judge our work.

I also thank all those who participated closely

Or from afar to develop this works.

Gift

I dedicate this work:

To my mother,

To my father

For my brother and sisters

For the whole family

And to all my friends

Safia

Abstract

The use of applications, such as social networking and e-commerce, is increasing exponentially, leading to an increase in content. Context-based recommendation systems play an important role in these applications.

It filters only relevant information from a large amount of available content. Content-based filtering (CBF) ***is the most widely used method for designing such a system. In recent years, deep learning has achieved great success across the board. However, its application in the field of recommendation is relatively new. In this work, this system helps to find the application that the user is looking for through MLP we have extracted the category from the description***

Abstrait

L'utilisation d'applications, telles que les réseaux sociaux et le commerce électronique, augmente de façon exponentielle, entraînant une augmentation du contenu. Les systèmes de recommandation contextuels jouent un rôle important dans ces applications.

Il filtre uniquement les informations pertinentes à partir d'une grande quantité de contenu disponible. Le filtrage basé sur le contenu est la méthode la plus largement utilisée pour concevoir un tel système. Ces dernières années, l'apprentissage en profondeur a connu un grand succès à tous les niveaux. Cependant, son application dans le domaine de la recommandation est relativement nouvelle. Dans ce travail, ce système aide à trouver l'application que l'utilisateur recherche à travers MLP nous avons extrait le contexte de la description

الملخص

يتزايد استخدام التطبيقات ، مثل الشبكات الاجتماعية والتجارة الإلكترونية ، بشكل كبير ، مما يؤدي إلى زيادة المحتوى. تلعب أنظمة التوصية المستندة إلى السياق دورًا مهمًا في هذه التطبيقات.

يقوم بتصفية المعلومات ذات الصلة فقط من كمية كبيرة من المحتوى المتاح. التصفية القائمة على المحتوى هي الطريقة الأكثر استخدامًا لتصميم مثل هذا النظام. في السنوات الأخيرة ، حقق التعلم العميق نجاحًا كبيرًا في جميع المجالات. ومع ذلك ، فإن تطبيقه في مجال التوصية جديد نسبيًا. في هذا العمل ، يساعد هذا النظام في العثور على التطبيق الذي يبحث عنه المستخدم من خلال سياق الوصف ، باستخدام تقنيات التعلم العميق

Contents

Contents

Contents

List of figures

List Of table

Abreviation list

RS: Recommender System

CBF: Content-based filtering

HM: Hybrid Methods

RS: Recommender System

CARS: Context Aware Recommender System

MLP: Multilayer neural networks

General Introduction

Due to the democratization of the internet and the exponential increase in the amount of information available and accessible, recommendation systems are becoming increasingly important.

However, they face a significant problem in a dynamic environment marked by changing user needs and the dynamic nature of available information content. In reality, pertinence, opportunity, and adaptability of resources sent to the needs and contexts of users are key success factors for recommendation systems.

Content-based Filtering, which is a general evolution of studies on information filtering, is based on the content of documents (topics covered) to compare them to a profile itself made up of themes. Each user of the system then has a profile that describes their own areas of interest. Two central functions stand out for a filtering system:

- The selection of documents relevant to the profile.

- Updating the profile according to the relevance feedback provided by the user on the documents he has received, the update is done by integrating the topics covered in the documents deemed relevant

Recommendation Systems are very powerful tools for suggesting services and objects that adapt to user profiles. However, a considerable amount of research done on recommendation systems focuses only on recommending the most relevant items to users without taking into account contextual information (location, time, etc.).

However, context is a very broad and vague concept which may vary depending on the application field. First, it must be acquired by possibly using sensors. Then, interpreted to obtain useful information, and modeled for simplify its use. Indeed, it is difficult to model the context from unobservable contextual information such as the intention of users. An alternative to solve this problem is to model themes from the textual data, which are a particularly rich and expressive source of information.

For the past ten years or so, neural models have taken on a great importance in many complex tasks thanks to algorithmic advances and the availability of powerful calculation tools such as

graphics processors. They are now very successful in many fields such as modeling, machine translation, recognition and prediction.

The main objective of our research is to develop a context-based recommendation system that adapts to user demand using a combination of deep neural networks and contextual recommendation systems to assist users in overcoming information overload.

Thus, we propose a context-sensitive recommender system based on a Multi-Layer Perceptron algorithm (MLP)

The proposed model uses text descriptions and a multi-layer neural network to extract the class to which the description belong .This implicit context is then integrated into a recommendation system to improve the quality of the predictions. This implicit context is then integrated into a recommendation system to improve the quality of the predictions.

This dissertation is organized into three chapters. The first one is devoted to the presentation of the basic concepts of Recommender Systems and deep learning. We mainly expose the recommendation methods, their limitations as well as evaluation metrics. We also focus on neural networks and their applications in recommender systems.

The second chapter is dedicated to a detailed conceptual study of the proposed model by presenting the main objective of our work, the architecture of our system and the different algorithms used. The last chapter presents the tools and languages used for the development of our recommendation system.

Finally, we end our brief with a general conclusion and some perspectives.

Chapter 1: State of the Art

I.1. INTRODUCTION:

Given the increase in the volume of information and the number of Internet users, it is becoming increasingly difficult to locate data, even when using traditional information search methods, which do not always produce appropriate results. However, recommendation algorithms have been created over the last ten years. This is what we will observe in this chapter as an excellent technique to lessen the complexity of information retrieval.

I.2.RECOMMENDER SYSTEMS:

Today, everyone is confronted with the problem of information overload, which is characterized by a vast array of options. However, choosing among a large number of possibilities in a short amount of time is tough. Recommender Systems (RSs) are collaborative, query-free systems that strive to recommend things, events, links, and information to individuals [1].

RS is typically a group of algorithms that learn and compute user preferences to uncover data patterns from a dataset. On the basis of the correlation between the interests and needs, the system then gives the most relevant and useful results. The primary goal of RS is to satisfy consumers and build long-term relationships with them. RS has grown in prominence in recent years as a result of their wide range of applications. In many existing RSs, a static user profile is the most common strategy, however it is insufficient to assess users' choices and preferences. The type of data has an impact on the technique used to construct a recommendation system. This information might be in the form of content, engagement (clicked data), or user information.

I.3.RECOMMENDATION METHODS:

A recommendation system makes it possible to extract information that is likely to be of interest to a particular user from the observation of their behavior and their possible assessments. A review of the literature in this area shows that there are several recommendation methods of which content-based filtering and collaborative and hybrid filtering are the most widely used [2].

I.3.1. Content-Based Filtering:

One of the most extensively utilized strategies in the recommendation system sector is

content-based recommendation. Based on the characteristics of the products, the recommendation system detects comparable relationships between them and then suggests more

goods that are similar to the ones they prefer. As a result, selecting the item's attributes is critical and strongly linked to the recommendation system's performance. .[2]

- The work in [3] proposed a recommendation system based on content. The Personalized Recommender System (acronym for Personalized Recommender System) generates dynamic hyperlinks for a website with a collection of personality enhancement advice. The data is processed by PRES, which sorts the elements into a positive category called C. (items relevant). He then determines the degree of similarity by looking at the relationships between item content and user preferences. PRES dynamically shows its suggestions by providing hypertext links to content pages containing the required things. It offers suggestions based on a comparison of each user. Each document in the collection has its own profile with its own content. The cosine measure in a vector space is used to determine the similarity between the profile vector and a document phrase, website content.

Collaborative Filtering:

By looking at his rating history and discovering similar users, the collaborative filtering algorithm locates the peer user. The neighborhood is then used by CF to generate the recommendation. A user-item matrix is typically required by CF Recommender Systems to reflect the users' ratings or comments on things. The ratings can be used to represent the preferences of users. After generating the matrix, the system will calculate the similarity between users in order to propose things to similar users, referred to as "neighbor users." Table 1 shows a user-item matrix, with the entries representing the users' ratings. The rates in this matrix are 0 and 1, and the rates can use more numbers to describe the various degrees of like or disliking. The structure of collaborative filtering systems is shown in

	Item 1	Item 2	Item 3	Item 3	……..	Item Y
User 1	0	1	0	1	……..	1
User 2	1	1	0	0	……..	0
User 3	1	0	1	1	……..	0
……..	……..	……..	……..	……..	……..	……..
User Y	1	0	0	1	……..	1

Table 1: Example of User-Item Matrix•

- The work in [4], presents a collaborative filtering-based recommendation system. They rely on information gleaned from a home improvement store's implicit database. They begin by categorizing items into categories like "tip." Following that, the product is handled as a generic pipe for each type of pipe. The developed recommendation system is designed to be used by a professional (in this case, a salesperson) to recommend acceptable products based on the system's mandated category We can't apply this system straight to a purchase history area where it's vital to recommend a specific product to a user without needing to utilize an expert because this method of recommendations is intended to be utilized by an expert and suggests only one category.
- The authors in [5] suggest a system that leverages a network of trust in addition to collaborative filtering with explicit information. The suggested system allows users to receive product recommendations based on the ratings of their trusted network. The system starts by taking a random trip around the user's trusted network in order to evaluate a product. If the product is discovered, the random walk is completed, and the rating is assigned as an estimate of the product's true rating. Otherwise, the march continues, with a chance that changes depending on the depth. The depth was set at 6 since the accuracy reduces as the search gets further away from the source. When walking comes to a halt, That is to say, no one was left standing. The system uses the rating of a similar product by calculating the Pearson correlation to see if anyone has quoted this product or whether the probability of halting has been attained.

Hybrid Methods:

Hybrid filtering methods combine multiple techniques to overcome the limitations of single filtering techniques. In order to tackle the cold-start or ramp-up difficulty as well as performance concerns, they usually combine collaborative and content-based approaches or mix collaborative and knowledge-based approaches. [6]

- They proposed in [7], merging content-based filtering and collaborative filtering with Markov Logic Networks to create hybrid models. Their research only used one form of probabilistic logic model, which is shown to be ineffective later in this study. Furthermore, it did not take into account the unique requirement of many recommendation systems, which is that precision be prioritized over recall.

An important notion in the RS field is the context, where contextual information may enhance the performance of the system and to generate pertinent recommendations.

1.4 RECOMMENDER SYSTEMS' PROBLEMS:

The use of referral systems has become a necessity, as they are able to suggest relevant resources with less effort and within a satisfactory response time. However, these systems suffer from some limitations, such as cold start, lack of ratings, selection of reliable neighbors, robustness and accuracy of recommendations, etc.

1.4.1 Cold start Problem:

This problem occurs when the system does not have enough data. The prediction algorithm will therefore be unable to generate recommendations. Cold start occurs for a new user starting with an empty profile, for a new item that may not be recommended, or for a new system where the data, on which the filtering process is based, is not available. [8].

1.4.2 Lack of data (sparsity problem)

This problem illustrates that the user × item matrix often has a small number of evaluations, which makes calculating similarities difficult. Several approaches have been proposed to alleviate the impact of lack of ratings on identifying reliable neighbors.

1.4.3 Reliable neighborhood identification (Neighbors training)

The neighborhood plays a very important role in a collaborative filtering system because the quality of the recommendations provided to the users is closely linked to the quality of the neighbors formed by the system. Statistical similarity measures, such as

Pearson's coefficient, allow us to identify the nearest neighbors based on the calculation of the similarities of the ratings between an active user and other users vis-à-vis co-rated items in the past.

I.4.3. Scalability

Recommendation systems generally contain a large amount of data. In collaborative filtering, the calculation of predictions uses a neighborhood of users or objects. Thus, with data that continues to grow, the time to compute similarities may increase dramatically, reducing system performance. It is therefore essential to use techniques allowing recommendation systems to make their suggestions quickly. Model-based methods overcome this problem by limiting the number of users or items used in the calculation of predictions.

I.4.4. Reliability

Referral systems remain vulnerable to malicious attacks and noisy data. Indeed, there is no guarantee that the assessments built into the systems actually reflect user preferences. There are many forms of attacks and noise. For example, some individuals may insert information in order to force the system to generate high marks for certain items [9].

I.4.5. Dynamicity

Collaborative filtering completely ignores changes in user habits depending on the context. It only takes into account user ratings expressed in the form of ratings [10]. Predicting a positive value does not in any way mean that the recommendation is good, you have to take the context and user needs into consideration.

I.4.6. Protection of privacy

Referral systems must ensure the protection of the user's personal information and ensure the separation of private and public data. On the one hand, this is to preserve the

anonymity of users and protect their information and on the other hand, to encrypt the transmitted data and perform information transfers over secure links.

Polat and Du [11] propose a collaborative filtering algorithm based on singular value decomposition. The proposed model uses specific communication protocols to ensure data confidentiality. The authors exploit random disruption techniques to protect privacy.

I.4.7. Precision of recommendations

Improving the accuracy of predictions is often a major issue in most of the research proposed within the framework of recommender systems [12]. Precision makes it possible to calculate the performance of the entire system. So, predict resources that meet the needs of users leading to their loyalty

. I.4.9. Novelty and diversity

The diversity of recommendations has a great influence on user satisfaction. Thus, a good recommendation system must find the best compromise to offer the user resources that are sufficiently similar to their known tastes without boring them with the same type of items [13]. This avoids redundancy in the suggested resources. Collaborative filtering provides some form of diversity. Thus, the problem of diversity particularly affects content-based methods.

I.5.Context Aware Recommender System (CARS)

The value of contextual information has been recognized by researchers in a variety of fields, including information retrieval, ubiquity computing, marketing and management, and so on. However, contextual information has been underutilized in several research projects on recommendation systems. Nonetheless, variables such as time, location, and the presence of other people can help to improve the recommendation process in some cases.

Traditional recommendation systems only deal with two sorts of entities: users and goods. However, for a variety of applications, For example, in tourism-related recommendation systems, it may not be sufficient to include only the users and the objects. It is frequently necessary to incorporate context information. For example, in order to provide appropriate vacation recommendations, a vacation recommendation system must take into account the season.In addition, a tourism recommendation system implemented on a mobile device

may prioritize the recommendation of activities close to the user's current location. Context is a broad notion for which a general and operational definition is particularly difficult. The The following definition, proposed by [14], is one of the most frequently accepted:

I.5.1Context Definition:

Context is a broad notion for which a general and operational definition is particularly difficult. The following definition, proposed by [14], is one of the most frequently accepted:

"Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves".

However, [14] have criticized this definition as being too broad and non-operational. The authors propose a context model based on [15], with five categories of contextual information: individuality, activity, time, relations, and location.

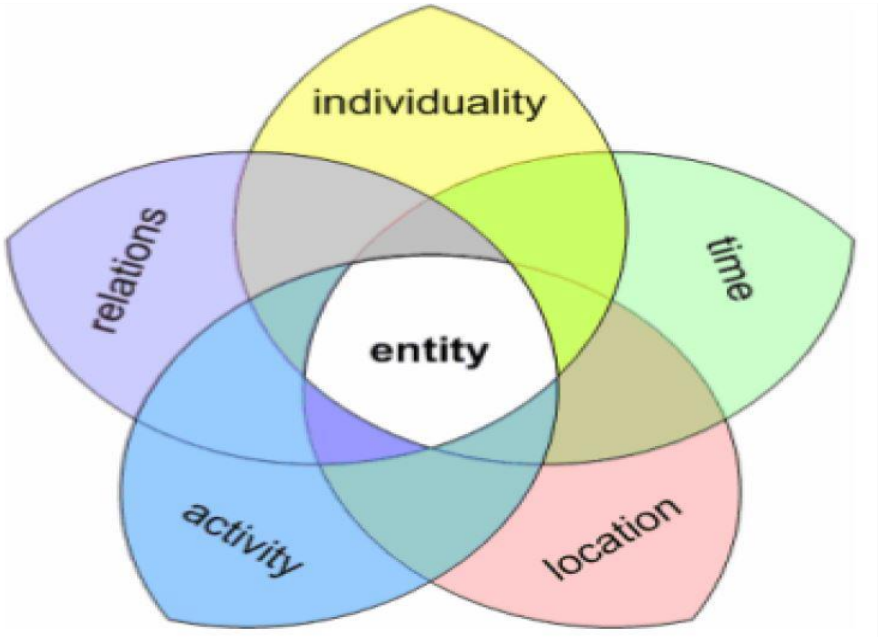


Figure I.1:The different elements of the context according to [14]

The context is defined in this way by [14]. *"Context is any information that can be used to characterize the situation of an entity. Elements for the description of this context information fall into five categories: individuality, activity, location, time, and relations."*

I.5.2.Contextual Information Sources

Context information can be achieved in three ways:

• Explicitly: by asking questions or utilizing other means, such as referring directly to the user or source who possesses the field information and receiving specific information.

• Implicitly: by relying on information or the environment, such as a change in user location obtained from a shared mobile operator. When a transaction is established, tags can also be used to acquire time field information.

• Deducing the field: Statistical approaches and data extraction are used. For instance, consider the age profile of someone who uses a video site and has an account. It may not be indicated officially in the system, but it can be properly estimated by looking at the videos the user has seen when he was a child.

I.5.3. modeling for recommendation systems:

Context modeling for recommendation systems There are three ways that contextual information can be used to improve recommendations:

1. Contextual pre-filtering (PreF): It implies that contextual information is utilized to filter away irrelevant ratings before they are employed in standard (non-contextual) methods to compute recommendations.

2. Contextual post-filtering (PoF): In this paradigm, contextual information is employed to generate recommendations after traditional non-contextual recommendation methods have been used.

3.Contextual modelling (CM): It presupposes that, in addition to the user and item data, contextual information is included in the recommendation-generating algorithms.

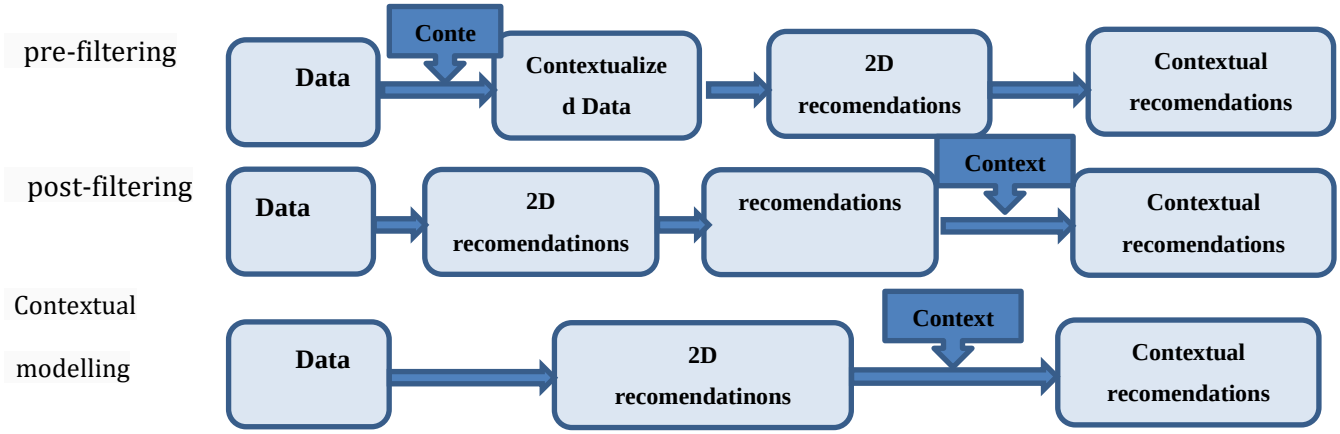


Figure I.2: How to interfere with context information in the proposed process in the traditional recommender system

Related work:

The work in [16] proposed a deep neural network architecture for learning the user-item interaction function. We create unique context-aware models that integrate the context dimension in order to learn nonlinear interactions between user, object, and context latent representations. For this goal, we propose several explicit and implicit context representations. Encoder-decoder neural networks (AE and LSTM encoder-decoder networks) are used to obtain the latent representations, which minimize the high-dimensional context space and show complicated relationships within the data.

The authors in [17]suggested a context-aware session-based RS based on conditional RNNs that injects contextual information into the input and output layers and alters the RNN's behavior by combining context embedding with item embedding. While their research supports the idea that contextual sequential modeling enhances recommendation accuracy, they only included three types of explicit contextual features in their study.

The work in [18] suggested Using RNN to anticipate the next item represented as a set of attributes, neural architecture developed an embedding of a sequence of occurrences. The contextual data was combined with the data for the associated items. In tests, the model based on events outperformed the model based just on objects.

Neural architecture proposed in [19] produced an embedding of sequence of events using RNN to predict the next item represented as a set of attributes. The contextual information was concatenated with associated item information. In experiments, the model with events context outperformed the model based only on items..

For their part, the researcher in [20]suggested a context-aware session-based RS based on conditional RNNs that injects contextual data into the input and output layers and alters the RNN's behavior by combining context embedding with item embedding. While their findings support the idea that contextual sequential modeling enhances suggestion accuracy, they do it in a different way. Only three sorts of explicit contextual information were included: time, time difference from the previous event, and event kind. We define context in our work as any user situation (physical and emotional) that may influence the user's preference. Many context dimensions (such as weather, sound, light, and location) are used. in order to extract latent contexts that aren't limited to a specific set of activities Because we presume there are many different sorts of contexts, modeling latent context is more difficult.

I.6.RECOMMENDER SYSTEMS AND NEURON NETWORKS:

An artificial neural network is a method of machine learning inspired by the biological model. It consists in mimicking human memorization and learning functions [21]. Thus, a deep neural network consists of thousands or even millions of neurons, divided into several dozen layers, each representing a processing step. In fact, the more the number of layers is increased, the more the networks of neurons learn complicated, abstract things, corresponding more and more to the way a human reasons.

Formally, a neural network is a directed and weighted graph. A node in the network represents a formal neuron which is an elementary computing unit with an activation value. Neurons are organized in layers and interconnected through weights that act as synaptic links. Each weight represents the degree of interconnection of the nodes it connects. A formal neuron is therefore characterized by (Figure I.2):

Learn more about this source text You must specify the source text for additional information. A set of influences from other neurons through synaptic links weighted.

- A weighted activation calculated from these influences.
- A threshold Θ which is a value below which the response of the neuron is ignored.
- Optionally, an external entrance allowing information to be expressed input neurons.

- An activation function that determines the actual activation of the neuron from its weighted activation, of its threshold, and of its external input.

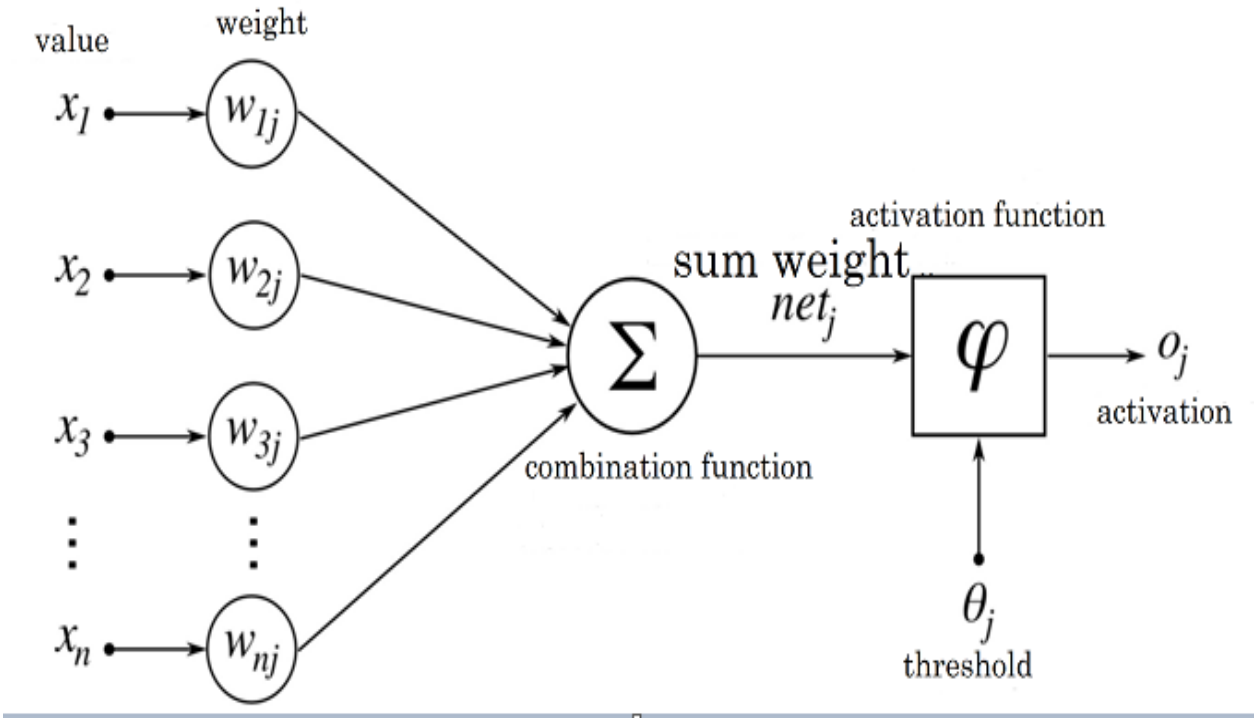


Figure I.3.. Structure of a formal neuron [W2]

Typically, a neuron receives a signal from the input layer or adjacent neurons, and in turn emits a signal representing its activation value when it is above a certain threshold.

A neural network can learn and generalize from representative examples.

This makes it possible to develop a model without having theoretical knowledge of the field. Thus, from a set of data, it is able to model a large variety of behaviors. It is also robust against noisy data. However, this technique has some drawbacks. In addition to the over-learning problem, it lack of method allowing to define the best topology of the network, the initial values of the weights, the learning step and the number of neurons of the different layers.

Indeed, these different parameters have a primordial role in the convergence of the network.

I.6.1.Training

The neural network generalizes knowledge from a database of examples. It adjusts its behavior using learning rules allowing the evolution of connection weights over time. Indeed, learning enables networks of neurons to perform complex tasks in different types of applications (classification, identification, character recognition, speech recognition, vision, control system, etc.). It is a gradual and iterative process based on the propagation of activation signals from input neurons to outputs. It consists of adjusting synaptic weights using a large amount of data, which is put together in a body of learning. Thus, the network receives as input data on which it applies a transform function to produce a result. The synaptic weights are then adjusted by minimizing an error function [21]. There are usually two types of learning: supervised learning and non-learning supervised. Supervised learning is the process of learning a function that, from a learning sample, best approximates the desired input-output relationship. The data used for supervised learning is a series of examples with pairs made up of input data and expected outputs. Unsupervised learning can find underlying structures from unlabeled data.

Thus, the system must detect similarities in the data it receives and organize them into groups or clusters. This technique is used for specific neural networks such as auto-encoder networks.

I.6.2.Deep Learning

Deep learning is a set of machine learning techniques based on artificial neural networks whose number of layers and

neurons is important. It allows machines to learn multiple levels of representations and abstractions of objects from large amounts of data. It is characterized by its considerable power and flexibility in learning to represent the world as a nested hierarchy of concepts. It is able to learn characteristics directly from the data, without having to perform manual extraction. Advances in deep learning have been made possible by the increase in the power of computers and the development of large databases (big data).

I.6.3.Multilayer neural networks (MLP)

The main idea of multilayer neural networks is to group neurons into layers. Several layers are then placed end to end and the neurons of two adjacent layers are completely

connected. The inputs of the neurons of the second layer are in fact the outputs of the neurons of the first layer. The neurons of the first layer are connected to the outside world and all receive the same input vector (this is inters the network) (Figure). They then calculate their outputs which are transmitted to the neurons of the second layer. The outputs of the neurons of the last layer form the output of the network. A multilayer neural network therefore calculates a vector function. The values of synaptic connections and thresholds can be adjusted to modify the calculated function.

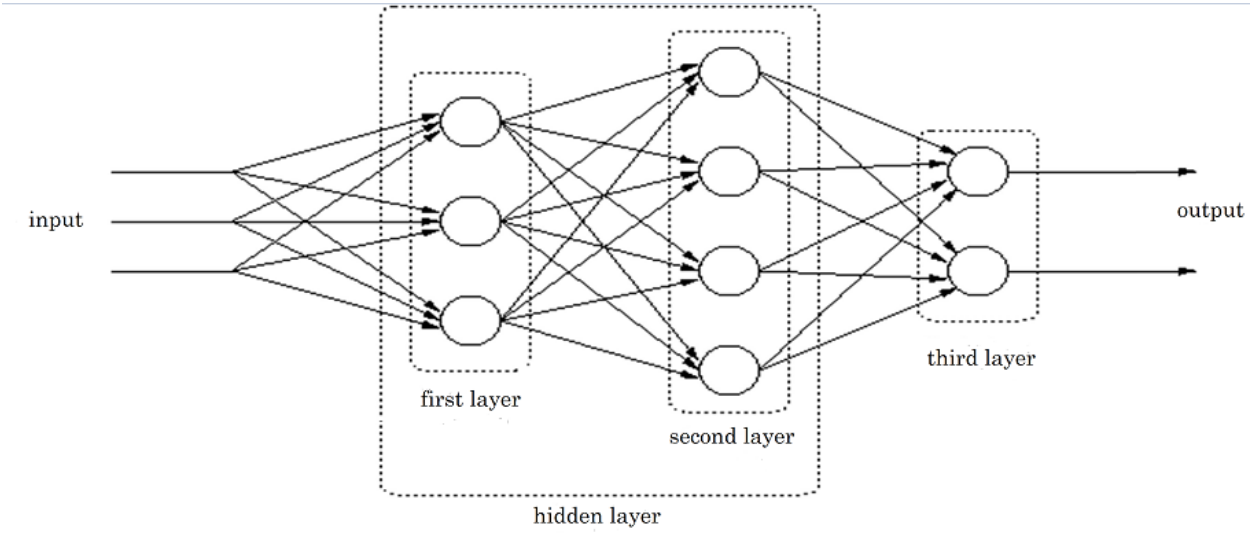


Figure I.4. Multilayer neural network

for example **[22]**, proposed Convolutional Matrix Factorization (ConvMF), which combines a CNN with Probabilistic Matrix Factorization to leverage information from user reviews for rating prediction.

In **[23]** used an RNN to learn representations from the textual content of scientific papers. Besides predicting ratings for a given article, they used multi-task learning to predict also item metadata such as genres or item tags from text.

In this chapter, we have discussed briefly the recommender systems field, the main recommendation techniques, and their limitations. We also presented the contextual recommendation systems as well as the various research works in this field. This work generally focuses on solving one or more problems among the issues facing this type of system.

I.7.CONCLUSION

In this chapter, we have discussed briefly the recommender systems field, the main recommendation techniques, and their limitations. We also presented the contextual recommendation systems as well as the various research works in this field. This work generally focuses on solving one or more problems among the issues facing this type of system.

Chapter 2: System Design

I. Introduction:

In this chapter, we'll pose a problem and try to solve it using the concept of network integrations and an approach that combines the Multilayer Perceptron (MLP) and a Content-based Filtering.

II. The Problematic:

Many present approaches to recommender systems focus on recommending the most relevant items to individual users without taking into account any contextual information, such as time, place, or other people's company (e.g., for watching movies or dining out). To put it another way, standard recommender systems deal with applications that only have two types of entities: people and objects, and they don't place them in context when making recommendations. However, in many applications, such as recommending a vacation package, personalized content on a Web site, or a movie, it may not be sufficient to consider only users and items—in order to recommend items to users under certain circumstances, it is also necessary to incorporate contextual information into the recommendation process. In a travel recommender system, for example, using the temporal context is very important to provide a vacation recommendation in the winter that can be very different from the one in the summer.

So, the following problematic arises:

How can implicit contextual information be captured and integrated into a content-based recommender system in order to improve the recommendations' quality?

III. Objective and motivation:

The main objective of our project is to capture and integrate the unobservable contextual information in a content-based recommender system by an automatic and intelligent method, to make the system more adaptive to the user's needs and preferences.

To achieve our objective, we propose to model the context, using unobservable contextual information, by capturing the implicit context from the user's description in the program, using a Multilayer Perceptron (MLP) based deep learning algorithm.

We have used this method because of its efficiency and performance proved in the domain over the state of the art presented in the previous chapter.

VI. Our Suggestion:

To find a solution to the previous problematic and to achieve our objectives, we propose in this thesis an adaptive context-aware recommender system that makes a recommendation based on the context extracted from the user's description. The system combines a deep learning-based Multilayer Neural Network (MLP) algorithm with a content-based one, to recommend items based on implicit context, and to help the recommender system improve the quality of recommendations.

In the next sections, we will present the details of the proposed approach.

VI.1. The system Architecture:

In **FigureII.5**, we present a general architecture of the proposed system that explains the operating process.

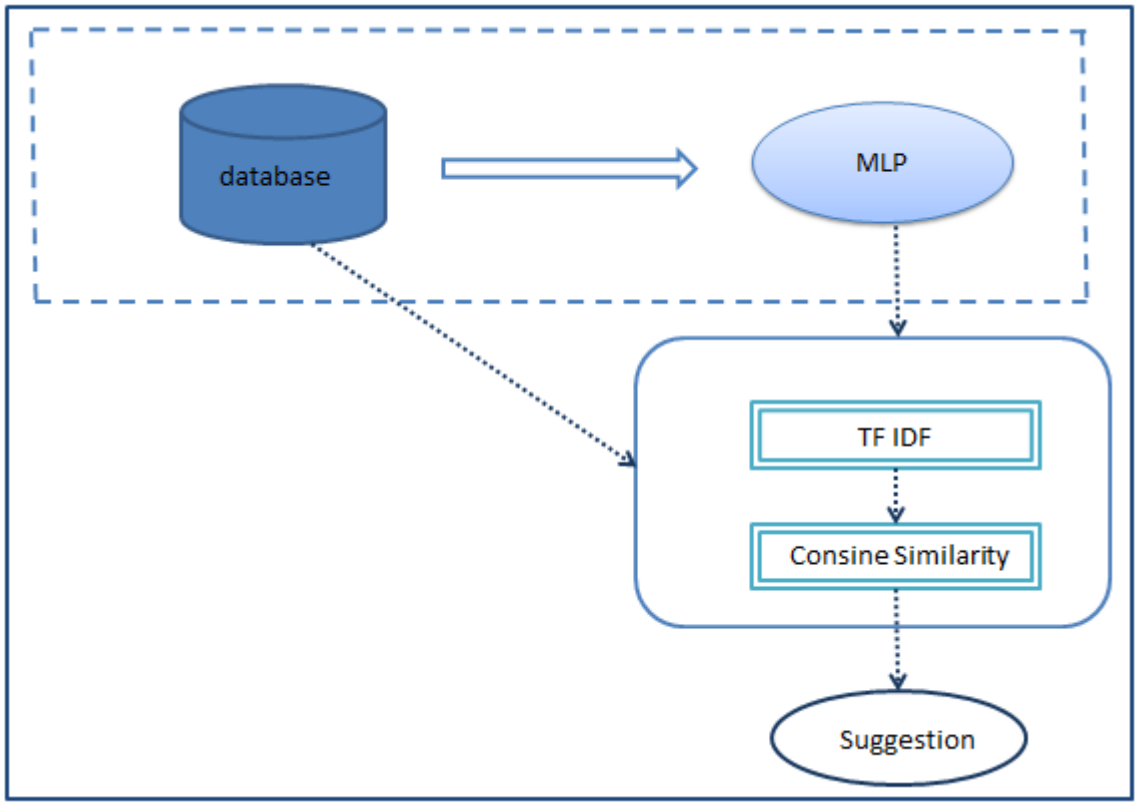


Figure II.1 /Architecture of the proposed system

The operating process begins when the user interacts with the system And enter the description of the application he wants

Then, the items' category is determined based on the description provided by the user using the MLP algorithm.

and then uses TF-IDF (Term Frequency-Inverse Document Frequency) algorithm and Cosine Similarity to recommend appropriate applications to the user

IV.2..Data Preparation

We processed the mobile_frappe.csv Which Contains 13column and 8361 rows, to fit the idea of our system

	GeneratedID	EntityNodeID	URL	HOST	s:name	s:description	
8346	-1319088329	nodef01e656cb8d39f...	http://yahoosports.tea...	yahoosports.teamfan...	Anastasio Moda NFL ...	Celebrate your Arizon...	Anastas
8347	-728981606	node6f54d55ee985c8...	http://yahoosports.tea...	yahoosports.teamfan...	New York Yankees La...	On those warm weath...	all you want to
8348	-222346879	nodeaf3bdcf1ad3e2cf...	http://yahoosports.tea...	yahoosports.teamfan...	Detroit Lions Wayfare...	The Lions roster is f...	as well as tear
8349	4791380	nodedc14e95fd8d492...	http://yahoosports.tea...	yahoosports.teamfan...	Nike Vision Tailwind1...	Protect your eyes and...	which feature
8350	224344007	noded5e3f894a3a86e...	http://yahoosports.tea...	yahoosports.teamfan...	Unisex Kansas City R...	Cheer on your Kansa...	Concep
8351	397822906	nodeea84c435d0eca...	http://yahoosports.tea...	yahoosports.teamfan...	Tampa Bay Buccanee...	Celebrate your Tamp...	perfect for any
8352	463605776	node71b6715d787c4...	http://yahoosports.tea...	yahoosports.teamfan...	Nike Baltimore Oriole...	Keep warm at the nex...	
8353	636899509	nodefbe08de9c01fbb...	http://yahoosports.tea...	yahoosports.teamfan...	Barcelona FC Nike H...	Cheer on the Barcelo...	and show a bit
8354	637037050	node6a13e0daafebd5...	http://yahoosports.tea...	yahoosports.teamfan...	adidas Hakeem Olaju...		
8355	743408990	node84781f5ff432541...	http://yahoosports.tea...	yahoosports.teamfan...	South Carolina Game...		
8356	1146044307	node5b54f4ab6ece87...	http://yahoosports.tea...	yahoosports.teamfan...	Women's Tampa Bay ...	Get in the game this s...	then look no fu
8357	1383891043	nodef594c9f4b219f32...	http://yahoosports.tea...	yahoosports.teamfan...	Jacksonville Jaguars ...	You know that you're...	but other fans
8358	1574867358	node41bd790c3fb34e...	http://yahoosports.tea...	yahoosports.teamfan...	Miami University Red...	Celebrate your fando...	
8359	1893432802	nodead43e09834ba5...	http://yahoosports.tea...	yahoosports.teamfan...	Nike Oakland Raiders...	As a die-hard NFL fan	you have a spe
8360	2038936445	nodefdc9aee521b0d4...	http://yahoosports.tea...	yahoosports.teamfan...	Baltimore Ravens La...	You go all out to dres...	lace trim and d
8361	2066166538	node632856c7167e8...	http://yahoosports.tea...	yahoosports.teamfan...	Mens Big & Tall St. Lo...	Celebrate your St. Lo...	as well as a tag

FigureII.2: mobile_frappe.csv

First we took the columns we needed using **df.dropna**, and we removed rows with the same name and description using **df.drop_duplicates** which become 3530 rows and 6 columns is shown in Figure ..

	item	name	description	category	developer	rating
0	0	Any.DO To-do & Tasks List	Meet Any.DO, the best way to-do To-do's on And...	Productivity	Any.DO	4.5
1	1	Yahoo!	With Yahoo! for Android, you'll stay connected...	News & Magazines	Yahoo! Inc.	4.2
2	2	Compass PRO	Professional Compass for Android. Simple and p...	Tools	Mobile Essentials	4.1
3	3	Instagram	Instagram – A beautiful way to share your worl...	Social	Instagram	4.6
4	4	Shopping List	Shopping List is the most easy way to organize...	Shopping	Kiwi3	4.5
...	...	...	...	...	...	...
4077	4077	Monster Galaxy Exile	Battle & tame hundreds of insane creatures in ...	Arcade & Action	Gaia Interactive	4.6
4078	4078	Legends of Yore	Legends of Yore, a casual rogue like designed ...	Casual	Coke and Code	4.5
4079	4079	Talking Jumpster from Mars	Talking Jumpster from Mars repeats everything ...	Brain & Puzzle	Moarbile Family	3.7
4080	4080	Extensive Notes Pro - Notepad	As seen on Lifehacker: "Extensive Notes Is a R...	Productivity	Fluffy Delusions	4.5
4081	4081	Android Pro Widgets	Advanced widgets package with many themes and ...	Productivity	Dr. Appche	4.4

3530 rows × 6 columns

FigureII.3: Database after deleting the replay

Then, we have organized the categories by appearance and converted them to numerical values. After that, we entered them into the **catDict table** that will be combined with the new database.

	item	name	description	category_y	category_x	rating	developer
0	0	Any.DO To-do & Tasks List	Meet Any.DO, the best way to-do To-do's on And...	0	Productivity	4.5	Any.DO
1	1	Yahoo!	With Yahoo! for Android, you'll stay connected...	1	News & Magazines	4.2	Yahoo! Inc.
2	2	Compass PRO	Professional Compass for Android. Simple and p...	2	Tools	4.1	Mobile Essentials
3	3	Instagram	Instagram – A beautiful way to share your worl...	3	Social	4.6	Instagram
4	4	Shopping List	Shopping List is the most easy way to organize...	4	Shopping	4.5	Kiwi3
...	...	...	...	...	...	...	...
3525	4077	Monster Galaxy Exile	Battle & tame hundreds of insane creatures in ...	10	Arcade & Action	4.6	Gaia Interactive
3526	4078	Legends of Yore	Legends of Yore, a casual rogue like designed ...	13	Casual	4.5	Coke and Code
3527	4079	Talking Jumpster from Mars	Talking Jumpster from Mars repeats everything ...	12	Brain & Puzzle	3.7	Moarbile Family
3528	4080	Extensive Notes Pro - Notepad	As seen on Lifehacker: "Extensive Notes Is a R...	0	Productivity	4.5	Fluffy Delusions
3529	4081	Android Pro Widgets	Advanced widgets package with many themes and ...	0	Productivity	4.4	Dr. Appche

3530 rows × 7 columns

Figure II.4 : Database after processing

VI.3.Application of the Multilayer Neural Network (MLP):

The design part of the learning of a neural network (the goal is to identify the class to which the user's description belongs) is combined with the digital calculation part of the learning of the network. MLP can have any number of layers, but in order to optimize its operation and minimize computation time as much as possible, the ideal design in terms of the number of layers and the number of neurons in each layer must be sought. The ideal weights are determined based on the chosen architecture and the available learning base.

VI.3.1General properties of an MLP:

The main properties of an MLP are:

- A neuron is connected in input to all the neurons of the next layer
- The output of a neuron is connected to all the neurons of the previous layer
- There is no connection between neurons in the same layer

- There is no looping from the outputs to the input
- The number of hidden layers is arbitrary, as well as the number of neurons in those hidden layers
- There is no loopback from the outputs to the inputs.
- The number of hidden layers is arbitrary, as well as the number of neurons in these hidden layers.

VI.3.2.Determine the number of input and output neurons

The network used for this work has an input layer, an output layer, and hidden (intermediate) layers. The number of neurons in the input layer depends on the number of descriptions. 3530 input neurons are used. The output is neurons representing the category which helps us in the next stage in the recommendation.

VI.3.3.Define the number of hidden layers

In addition to the input and output layers, the number of intermediate layers must also be determined. Indeed, without hidden layers, the adaptation of the network will be very limited.

Set the number of neurons for each hidden layer

Each additional neuron takes into account specific characteristics of the input neurons. It is therefore sufficient to add a sufficient number of neurons at the level of each hidden layer to adapt the network to our problem.

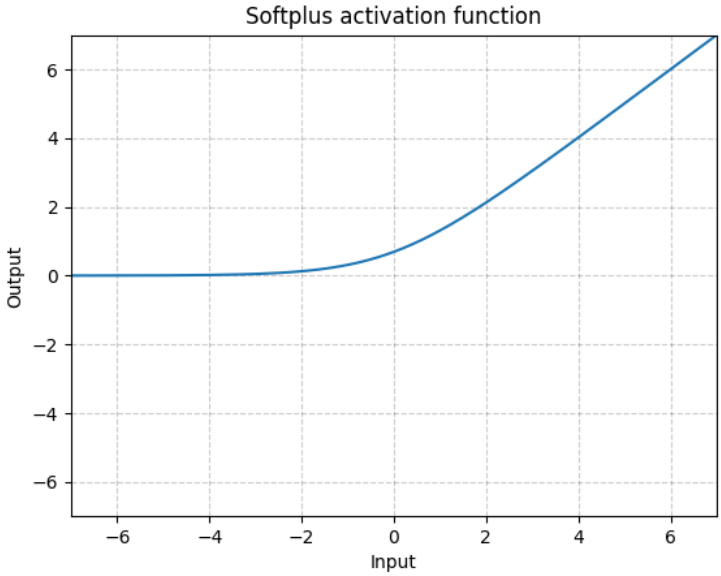
VI.3.4.Choose the activation function

In the context of this work, we consider the softplus function for the passage from the input layer to the intermediate layers, and for the passage from the hidden layer to the output layer.

softplus is a smooth approximation of the ReLU function and can be used to force the output of a machine to be always positive.

$$\text{softplus(x)} = \frac{\beta}{1} * log(1 + exp(\beta * x) \qquad (01)$$

The function will look more like the ReLU function, if the betaβ gets bigger and bigger.



FigureII.5:Softplus activation function

VI.3.5.Network learning

Because we wish to instill a specific benchmark behavior in the network, learning is supervised. As a result, the network is capable of detecting differences in its reference behavior and changing its weights to reduce this error.

The network is given the input data set and requested to alter its weights in order to get the desired output: the first stage is to propagate the inputs forward until the network calculates an output. After that, it will be compared to the desired result. The network weights will be adjusted as a result, and in the next iteration, the difference between the calculated and intended outputs is kept to a minimum. To determine the contribution of the error to the change in synaptic weights, it is propagated backward to the input layer. Finding the coefficients that minimize an overall error function is defined as an optimization challenge when learning the MLP network.

VI.3.6. TF-IDF and Cosine Similarity ：

We trained the model and extracted the category, but where is the android app recommendation system? Now that we have the category, we can use them to recommend which apps our model has learned are the most similar to a given app and so we TF-IDF and Cosine Similarity. The two basic concepts used in this recommendation system are Term frequency- Inverse Document frequency (TF-IDF) and Cosine Similarity. TF-IDF is

used for the to create vectors of the data and cosine similarity is used to compute the similarity measure between the vectors

V.4.Term Frequency-Inverse Document Frequency

This method is commonly used as a part of content-based recommendations systems. It consists of two terms. They are Term Frequency (TF) and Inverse Document Frequency(IDF). Term Frequency deals with the frequency of interests and favorites in user profile. Whereas, Inverse document frequency deals with inverse of term frequency among the whole data provided by user profile. These two concepts are combined together in order to provide the recommendation for a user based on the data's provided by user profile.

V.5.Cosine Similarity

Cosine Similarity is a complex concept which has been widely discussed in information retrieval. This algorithm converts a text document as a vector of terms. By this model, the similarity between two vector can be found by determining cosine value between two vectors. Application of this algorithm can be performed on any two texts such as documents, sentence or paragraph .Sometimes during the similarity measurement between the vectors yields unstable results. In case of search engines, the similarity value between user query and documents are determined and then it is categorized from highest to lowest one. Higher the similarity score between the user query vector and document vector means more relevancy between query and document.

V.6. Implementation Steps:

Step 01: Loading word_tokenize, stop_words and TfidfVectorizer libraries

```
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from sklearn.feature_extraction.text import TfidfVectorizer
```

FigureII.6: used libraries

Step 02: We use the function dataTrainTest for test data is 25%

```
def dataTrainTest(t,target,test_size=0.25):
    from sklearn.model_selection import train_test_split
```

FigureII.7: TrainTest function

Step 03: use the function mlpTrain to train the model

```
def mlpTrain(x_train,y_train,activation = 'relu'):
```

FigureII.8: mlpTrain function

Step 04: after train the model we use the function findCategory

```
def findCategory(entredString):
    [clf,tf] = joblib.load('trainedModel.sav')
```

FigureII.9: find Category function

Step 05: Now that we have our models, we need to find what the best number of application . As stated before, we'll be using cosine Similarity

$$\text{Similarity} = \text{COS}(\theta) = \frac{A.B}{||A||.||B||}$$

Step 06 : Create TF-IDF model

First, we'll set up Tfidf Vectorizer and tell it to use English stop words. This will remove common words like "the" and "of" to leave the more important ones. TF-IDF will additionally down-weight common words that appear across documents.

```
#Generate tfidf matrix model for entire corpus
tfidf_matrix = TfidfVectorizer(stop_words='english', min_df=2)
```

FigureII.10: TF-IDF Vectorizer

Next, we'll create a TF-IDF matrix by passing the text column to the fit_transform() function. That will give us the numbers from which we can calculate similarities.

```
data_tfidf_matrix = tfidf_matrix.fit_transform(cleaned_data)
pk.dump(data_tfidf_matrix,open('data/data_tfidf_matrix.pkl','wb'))
pk.dump(tfidf_matrix,open('data/tfidf_matrix.pkl','wb'))
return data_tfidf_matrix
```

FigureII.11: TF-IDF matrix

Step 07 : Now the model is built, and we have our TF-IDF matrix and a cosine similarity matrix covering all the descriptions, we can create a function to generate content recommendations

```
def getRecommandation(user_data):
    data_tfidf_matrix = pk.load(open('data/data_tfidf_matrix.pkl','rb'))
    tfidf_matrix = pk.load(open('data/tfidf_matrix.pkl','rb'))
    user_data_tfidf_vector = tfidf_matrix.transform([user_data])
    user_data_tfidf_vector.toarray()
    similarity_score = cosine_similarity(data_tfidf_matrix, user_data_tfidf_vector)
    recommendation_id = similarity_score.flatten().argsort()[::-1]
```

FigureII.12: get Recommandation function

Step8: now for the recommender we use two methods

- The first is add category to the entred string

```
df,data = readCsv()
generate_save_tfidf_matrix()
entredString1 = entredString+' ' + findCategory(entredString)
```

FigureII.13: The first methods

- The second method consists of filtering the dataset by category, than finding similarity with entered string:

```
catClass = findCategory(entredString)
df,data = readCsv(catClass)
generate_save_tfidf_matrix(catClass)
```

FigureII.14: the second methods

V.Conclusion:

In this second chapter, we have presented the proposed approach and its architecture and we have also presented in detail the design of our contextual recommendation system. The next chapter presents the implementation and operation of our application

Chapter 3: Realization and Implementation

III.1. INTRODUCTION:

In this chapter, we present the implementation of the proposed approach within the framework of a recommendation system based on deep learning, presenting the tools used as well as the experimental results. We start first with an overview of resources, language and development environment.

III.2. Hardware used:

To build our system, we used the following hardware configuration:

- PC: Hp Processor (Intel (R) Celeron (R)CPU N3060 @ 1.6GHz (2 CPUs)).
- RAM: 4.00 GB.
- Operating System: Windows 10 Professional 64-bit.

III.3. Dataset:

We used a Mobile_Frappe 2015 dataset, in application. This dataset was retrieved from website: *github.com* and includes 8361 rows.

III.4. Programming language and platform:

III.4.1. Python Program:

Python is an interpreted, multi-paradigm, cross-platform programming language. It promotes structured, functional and object-oriented imperative programming. It has strong dynamic typing, automatic memory management by garbage collection and an exception management system; it is thus similar to Perl, Ruby, Scheme, Small talk and Tcl.

Python is dynamically typed and retrieved. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python is often described as a "stacks included" language because of its extensive standard library. [24]

Why Python?

- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc.).
- Python has a simple syntax similar to the English language.
- Python has a syntax that allows developers to write programs with fewer lines than some other programming languages.

- Python runs on an interpreter system, which means code can be executed as soon as it's written. This means that prototyping can be very fast.
- Python can be handled procedurally, object-oriented, or functionally
- Platform Use: The platform used for our experiments is the Fedora V29 operating system.
- Fedora (GNU / Linux).

III.4.2. JupyterLab

Jupyter's Next-Generation Notebook Interface JupyterLab is a web-based interactive development environment for Jupyter notebooks, code, and data. JupyterLab is flexible: configure and arrange the user interface to support a wide range of workflows in data science, scientific computing, and machine learning. JupyterLab is extensible and modular: write plugins that add new components and integrate with existing ones.[25]

III.4.3. Libraries used

In order to implement the proposed model, we have used several python libraries like:

- Pandas
- numpy
- joblib
- MLPClassifier, etc.

III.4.4. MLPClassifier

MLPClassifier stands for Multi-layer Perceptron classifier which in the name itself connects to a Neural Network. Unlike other classification algorithms such as Support Vectors or Naive Bayes Classifier, MLPClassifier relies on an underlying Neural Network to perform the task of classification.

III.5. Description of functionalities

Once an application's description is entered, we find the category of the application, we are looking for, with the multi-layer perceptron (MLP) algorithm as follow:

```
findCategory('elegance and beauty')
```

```
<1x47557 sparse matrix of type '<class 'numpy.float64'>'
        with 3 stored elements in Compressed Sparse Row format>
'Lifestyle'
```

Figure III.1: find category

- **The first** method supports recommendation with category

```
<1x47557 sparse matrix of type '<class 'numpy.float64'>'
       with 4 stored elements in Compressed Sparse Row format>
elegance and beauty & Lifestyle   Lifestyle
[3529, 1172, 1183, 1182, 1181, 1180, 1179]
```

	item	name	description	category_y	category_x	rating	developer
3529	4081	Android Pro Widgets	Advanced widgets package with many themes and ...	0	Productivity	4.4	Dr. Appche
1172	1334	Textra SMS	Let's face it the stock Android Messaging app ...	5	Communication	4.3	Delicious Inc.
1183	1348	Met Office Weather Application	As the UK's national meteorological service, t...	21	Weather	4.0	Met Office
1182	1347	Gun Shot Sounds	Want to feel the force of a powerful virtual w...	9	Entertainment	4.0	Mob Buddy
1181	1346	Animals memory game for kids	* The animals memory game for kids is a game f...	12	Brain & Puzzle	4.6	Droid Development
1180	1345	Bubble level	A spirit level. Hold any of the phone's four s...	2	Tools	4.4	Antoine Vianey
1179	1344	Breaking News	Breaking News tells you what's happening right...	1	News & Magazines	4.0	NBC News Digital LLC

Figure III 2: Recommendation generated by the first method

- The second method recommends only applications of the category obtained. Results of the Top-N recommendation are presented in Figure …

```
<1x47557 sparse matrix of type '<class 'numpy.float64'>'
        with 4 stored elements in Compressed Sparse Row format>
elegance and beauty & Lifestyle   Lifestyle
[3529, 1172, 1183, 1182, 1181, 1180, 1179]
```

	item	name	description	category_y	category_x	rating	developer
3529	4081	Android Pro Widgets	Advanced widgets package with many themes and ...	0	Productivity	4.4	Dr. Appche
1172	1334	Textra SMS	Let's face it the stock Android Messaging app ...	5	Communication	4.3	Delicious Inc.
1183	1348	Met Office Weather Application	As the UK's national meteorological service, t...	21	Weather	4.0	Met Office
1182	1347	Gun Shot Sounds	Want to feel the force of a powerful virtual w...	9	Entertainment	4.0	Mob Buddy
1181	1346	Animals memory game for kids	* The animals memory game for kids is a game f...	12	Brain & Puzzle	4.6	Droid Development
1180	1345	Bubble level	A spirit level. Hold any of the phone's four s...	2	Tools	4.4	Antoine Vianey
1179	1344	Breaking News	Breaking News tells you what's happening right...	1	News & Magazines	4.0	NBC News Digital LLC

Figure III 3: Recommendation generated by the second method

We note that the recommendations resulting from the second method are better than the first one.

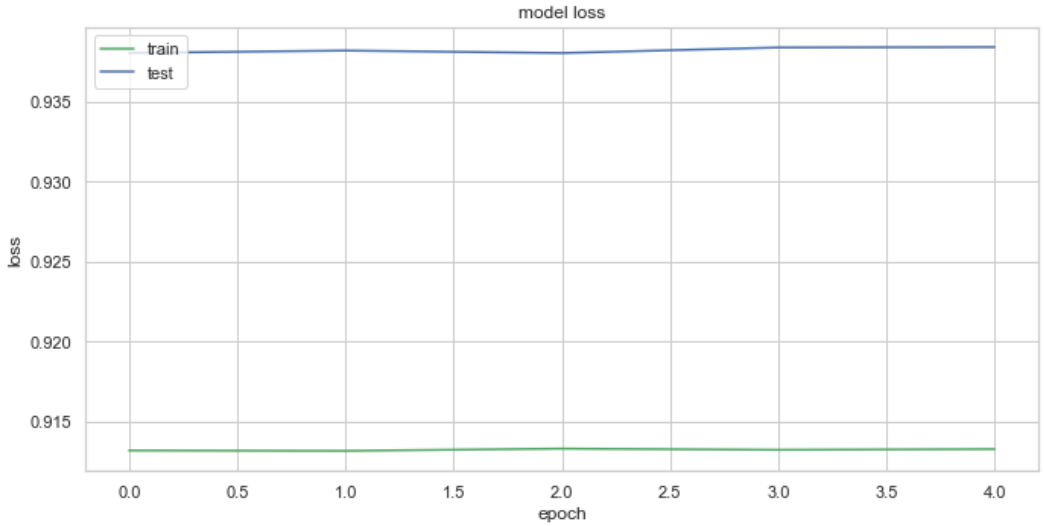
III.6. .Results obtained and discussion with MLP learning :

learning and testing the neurons network of our system. The tests concern the verification of the performance of the neuron network. Indeed, this validation is expressed by the value Loss (Loss) and Loss Validation (Val_Loss)

- **With 05 Epoch of learning:**

  We notice from the results obtained in the 5 epochs of learning test that the loss function started at 0.9132 and the precision at 0.9380 and after 5 epochs of learning the loss is 0.9133 and the precision is equal to 0.9384.

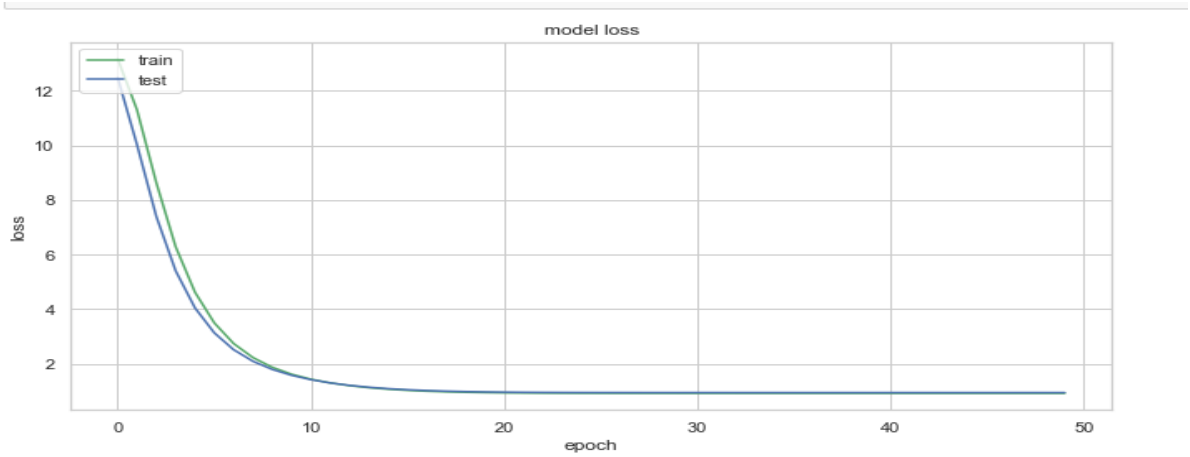
```
Epoch 1/5
625/625 [==============================] - 1s 2ms/step - loss: 0.9132 - val_loss: 0.9380
Epoch 2/5
625/625 [==============================] - 1s 2ms/step - loss: 0.9132 - val_loss: 0.9382
Epoch 3/5
625/625 [==============================] - 1s 2ms/step - loss: 0.9133 - val_loss: 0.9380
Epoch 4/5
625/625 [==============================] - 1s 2ms/step - loss: 0.9132 - val_loss: 0.9384
Epoch 5/5
625/625 [==============================] - 1s 2ms/step - loss: 0.9133 - val_loss: 0.9384
```



- **With 50 epoch of learning:**

We notice from the results obtained in the test of 5 epochs of learning that the loss function started at 13.44 and the precision at 12.49 and after 5 epochs of learning the loss is 0.7133 and the precision is equal to 0.7383.

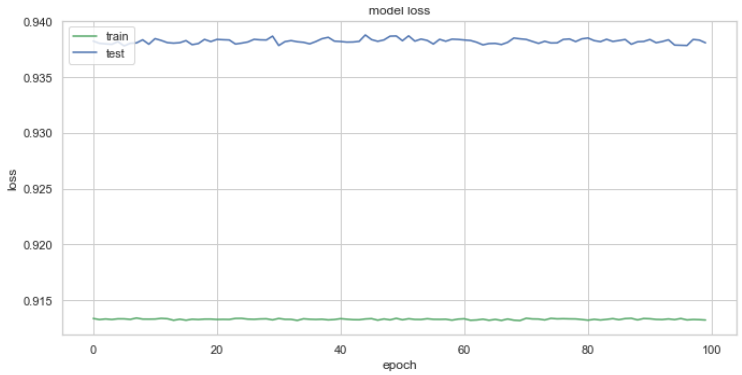
```
Epoch 1/50
625/625 [==============================] - 17s 3ms/step - loss: 13.4408 - val_loss: 12.4949
Epoch 2/50
625/625 [==============================] - 1s 2ms/step - loss: 11.8889 - val_loss: 10.0322
Epoch 3/50
625/625 [==============================] - 2s 2ms/step - loss: 9.2763 - val_loss: 7.3995
Epoch 4/50
625/625 [==============================] - 1s 2ms/step - loss: 6.8205 - val_loss: 5.4064
Epoch 5/50
625/625 [==============================] - 1s 2ms/step - loss: 4.9718 - val_loss: 4.0453
Epoch 6/50
625/625 [==============================] - 1s 2ms/step - loss: 3.7617 - val_loss: 3.1298
Epoch 7/50
625/625 [==============================] - 1s 2ms/step - loss: 2.9142 - val_loss: 2.5122
Epoch 8/50
625/625 [==============================] - 1s 2ms/step - loss: 2.3151 - val_loss: 2.0873
Epoch 9/50
625/625 [==============================] - 1s 2ms/step - loss: 1.9380 - val_loss: 1.7886
Epoch 10/50
625/625 [==============================] - 1s 2ms/step - loss: 1.6814 - val_loss: 1.5724
Epoch 11/50
625/625 [==============================] - 1s 2ms/step - loss: 1.4586 - val_loss: 1.4129
Epoch 12/50
625/625 [==============================] - 1s 2ms/step - loss: 1.3030 - val_loss: 1.2932
Epoch 13/50
625/625 [==============================] - 1s 2ms/step - loss: 1.2106 - val_loss: 1.2032
Epoch 14/50
625/625 [==============================] - 2s 2ms/step - loss: 1.1304 - val_loss: 1.1345
Epoch 15/50
625/625 [==============================] - 1s 2ms/step - loss: 1.0827 - val_loss: 1.0822
Epoch 16/50
625/625 [==============================] - 1s 2ms/step - loss: 1.0258 - val_loss: 1.0428
Epoch 17/50
625/625 [==============================] - 1s 2ms/step - loss: 0.9921 - val_loss: 1.0132
Epoch 18/50
625/625 [==============================] - 1s 2ms/step - loss: 0.9713 - val_loss: 0.9914
Epoch 19/50
625/625 [==============================] - 1s 2ms/step - loss: 0.9477 - val_loss: 0.9747
Epoch 20/50
625/625 [==============================] - 1s 2ms/step - loss: 0.9324 - val_loss: 0.9633
Epoch 21/50
625/625 [==============================] - 1s 2ms/step - loss: 0.9204 - val_loss: 0.9550
Epoch 22/50
625/625 [==============================] - 1s 2ms/step - loss: 0.9183 - val_loss: 0.9491
```



- **With 100 epoch of learning:**

We notice from the results obtained in the test of 5 epochs of learning that the loss function started at 0.9134 and the precision at 0.9371 and after 5 epochs of training the loss is 0.9132 and the precision is equal to 0.9372.

```
625/625 [==============================] - 1s 2ms/step - loss: 0.9134 - val_loss: 0.9379
Epoch 89/100
625/625 [==============================] - 1s 2ms/step - loss: 0.9132 - val_loss: 0.9382
Epoch 90/100
625/625 [==============================] - 1s 2ms/step - loss: 0.9134 - val_loss: 0.9382
Epoch 91/100
625/625 [==============================] - 1s 2ms/step - loss: 0.9133 - val_loss: 0.9384
Epoch 92/100
625/625 [==============================] - 1s 2ms/step - loss: 0.9133 - val_loss: 0.9381
Epoch 93/100
625/625 [==============================] - 1s 2ms/step - loss: 0.9133 - val_loss: 0.9382
Epoch 94/100
625/625 [==============================] - 1s 2ms/step - loss: 0.9133 - val_loss: 0.9383
Epoch 95/100
625/625 [==============================] - 1s 2ms/step - loss: 0.9133 - val_loss: 0.9379
Epoch 96/100
625/625 [==============================] - 1s 2ms/step - loss: 0.9134 - val_loss: 0.9379
Epoch 97/100
625/625 [==============================] - 1s 2ms/step - loss: 0.9132 - val_loss: 0.9378
```



- **With 150 epoch of learning:**

We notice from the results obtained in the test of 5 epochs of learning that the loss function started at 0.9133 and the precision at 0.9379 and after 5 epochs of training the loss is 0.9133 and the precision is equal to 0.9383.

```
Epoch 142/150
625/625 [==============================] - 1s 2ms/step - loss: 0.9133 - val_loss: 0.9377
Epoch 143/150
625/625 [==============================] - 1s 2ms/step - loss: 0.9133 - val_loss: 0.9377
Epoch 144/150
625/625 [==============================] - 2s 2ms/step - loss: 0.9132 - val_loss: 0.9380
Epoch 145/150
625/625 [==============================] - 1s 2ms/step - loss: 0.9133 - val_loss: 0.9383
Epoch 146/150
625/625 [==============================] - 1s 2ms/step - loss: 0.9133 - val_loss: 0.9383
Epoch 147/150
625/625 [==============================] - 1s 2ms/step - loss: 0.9133 - val_loss: 0.9379
Epoch 148/150
625/625 [==============================] - 1s 2ms/step - loss: 0.9133 - val_loss: 0.9381
Epoch 149/150
625/625 [==============================] - 1s 2ms/step - loss: 0.9133 - val_loss: 0.9382
Epoch 150/150
```

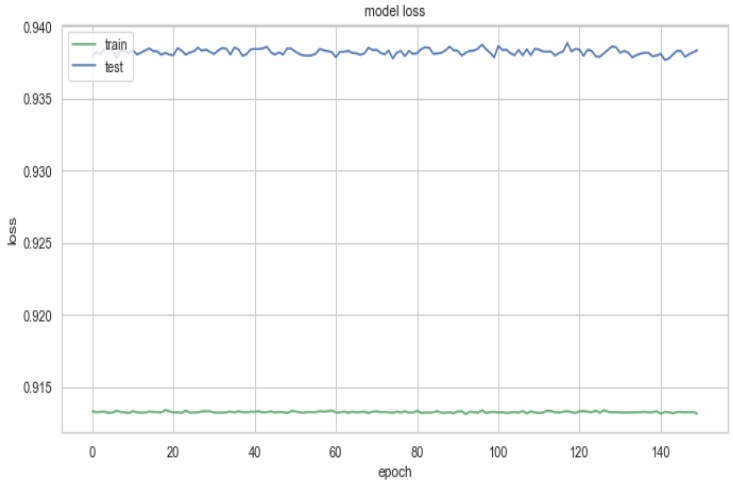


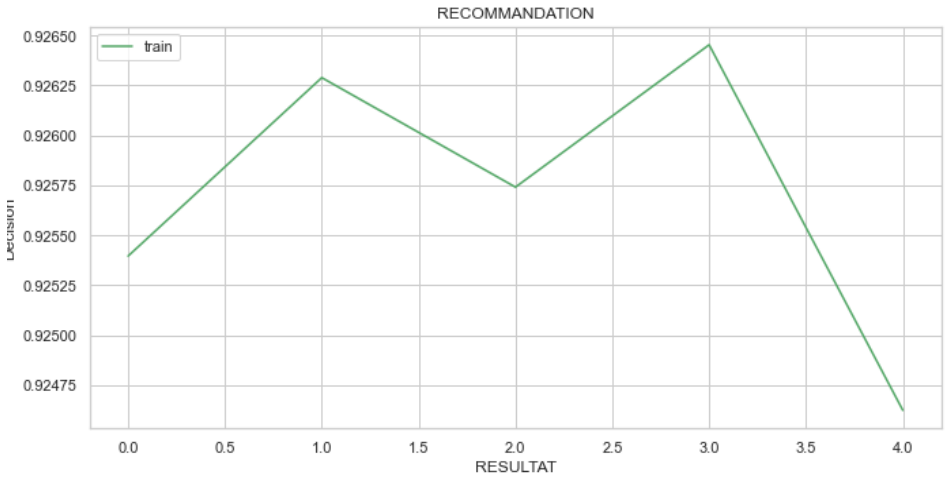
Table III.7 shows the main results obtained during the test phase for different epoch (5, 20, 50 and 100 and 150). It illustrates the results obtained at the beginning and at the end of the epoch.

Table III.1. represents the main results obtained during the test phase for different time epoch (5, , 50, 100 and 150). It illustrates the results obtained at the beginning and at the end of the epoch.

	Loss		**Val-Loss**	
5 epoch	**09132**	**0.9133**	**0.9380**	**0.9384**
50epoch	**13.44**	**0.7133**	**1249**	**0.7338**
100 epoch	**0.9134**	**0.9132**	**0.9371**	**0.9372**
150 epoch	**0.9133**	**0.9133**	**0.9379**	**0.9383**

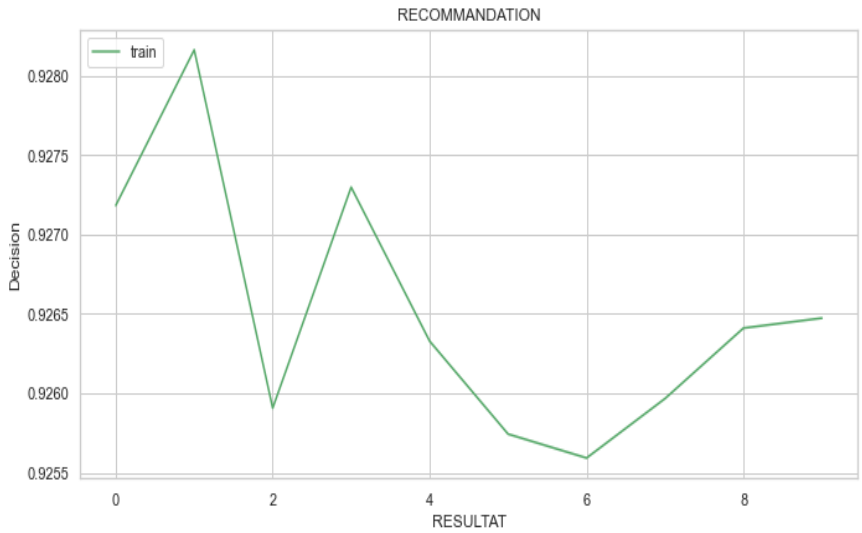
Table III.1. Principle results of MLP

III.7. Evaluation of the performance of the recommendation:



Top-5RECOMMENDATION :

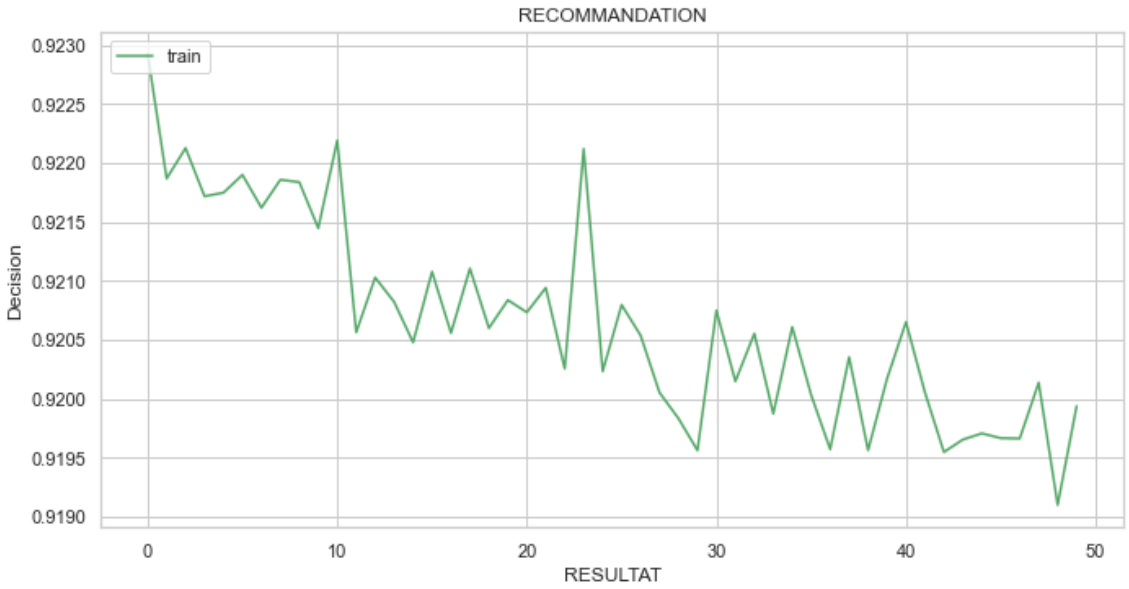
We note that the best recommendation is 0.92648 in Recommendation number 03.



Top-10RECOMMENDATION :

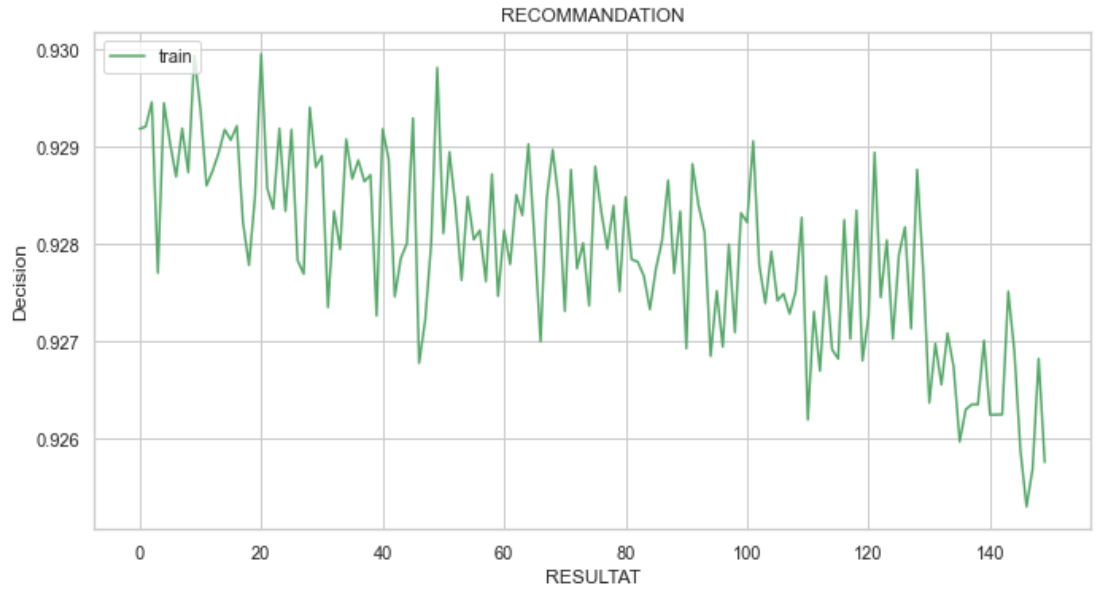
We note that the best recommendation is 0.9280 in Recommendation number 1:

.



Top-50RECOMMENDATION

We note that the best recommendation is 0.9230 in Recommendation number 0.



- **Top-150RECOMMENDATION:**

We note that the best recommendation is 0.930 in Recommendation number 20.

	Best recommendation	value
Top-5RECOMMENDATION	03	0.92648
Top-10RECOMMENDATION	01	0.9280
Top-50RECOMMENDATION	50	0.9230
Top-150RECOMMENDATION	20	0.930

Table III.2; Principle results of recommendation

III.7.Conclusion

This chapter describes the implementation of the proposed system able of taking into account unobservable contextual information in a content-based scheme.

Results were promising and showed that the second method is better than the first one to generate pertinent recommendations.

General Conclusion and Perspectives

With the explosive growth of information available on the Web, recommendation systems have becoming an indispensable tool for suggesting relevant information to users that suits their choices and preferences.

Recently, due to the powerful capabilities of machine learning for representation, its methods have been successfully applied in various fields such as e-commerce Web sites and / or media and also social networks. Much effort has also been made to apply deep learning models in recommender systems, to enhance their performance and improve the recommendation's quality.

In this thesis, we have presented a state of the art on contextual recommendation systems (approaches, problems, evaluation metrics, etc.) and their application, while highlighting methods using Multi layer Perceptron.

Next, we have proposed an adaptive contextual recommendation system that combines an MLP algorithm with a content-based one. The proposed system generates recommendations for users that fit their needs, to find the Android applications, only by entering the description of an application. The model uses the implicit contextual information extracted from this description, and deduces the category to which it belongs. Then, the recommendation is made in two ways. In the first one, by using the description of an application in addition to the category predicted, and in the second one, by considering just the applications that belong to the category extracted.

Results of experimentations were promising and showed an enhancement in the recommendation quality, when taking the contextual information in the account.

The performance of recommender systems depends from a dataset to another, so as a future work, we aim testing the proposed method on another dataset with different contextual information.

[1]J. Bobadilla , F. Ortega, A. Hernando, A. Gutiérrez Recommender systems survey 2013

[2] Robin van Meteren and Maarten van Someren Using Content-Based Filtering for Recommendation

[3] Zhi Li1 & Xiaozhu Zou, A Review on Personalized Academic Paper Recommendation,2019
A_Review_on_Personalized_Academic_Paper_Recommenda.pdf

[4] Julien Delporte, Céline Rouveirol ,Sebastien Guerif ,Francoise Soulie Fogelman A case study in a recommender system based on purchase data August 2011

[5] Mohsen Jamali , Martin Ester,TrustWalker: A Random Walk Model for Combining Trust-based and Item-based Recommendation

[6] Raza Ul Haq , Hybrid Recommender System towards User Satisfaction, University of Ottawa April, 2013

[7] Julia Hoxha and Achim Rettinger, First-Order Probabilistic Model for Hybrid

[8] Schein A. I., Popescul A., Ungar L. H., Pennock D. M., "Generative Models for Cold- Start Recommendations", In: Proceedings of the 2001 SIGIR Workshop on RecommenderSystems, 2001.

[9] O'Donovan J., Smyth B., "Trust in recommender systems", In: Proceedings of the 10thinternational ACM conference on Intelligent user interfaces, New York, USA, pp. 167 - 174, 2005.

[10] Gasmi I., Seridi-Bouchlaghem H., Labar H., "Collaborative filtering recommendation based on research habit of users", In: Proceedings of the third IEEE International Conference on Multimedia Computing and Systems, Tangier, Morocco, 2012

[11] Polat H., Du W., "SVD-based collaborative filtering with privacy", In: Proceedings of the ACM Symposium on Applied Computing, 2004.

[12] Pazzani M. J., "A framework for collaborative, content-based and demographic filtering", Artificial Intelligence Review, Vol.13, N°. 5, pp. 393 - 408, 1999.

[13] Jamali M., Ester M., "Using a trust network to improve top-n recommendation", In:Proceedings of the third ACM conference on Recommender systems (RecSys '09), New York, USA, pp. 181 - 188, 2009.

[14] Zimmermann, A., Lorenz, A., and Oppermann, R. (2007). An operational definition of context. In International and Interdisciplinary Conference on Modeling and Using Context, pages 558–571. Springer.Recommendations Article · July 2013

[15] Abowd, G. D., Dey, A. K., Brown, P. J., Davies, N., Smith, M., and Steggles, P. (1999). Towards a better understanding of context and contextawareness. In Handheld and ubiquitous computing, pages 304–307. Springer

[16] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In Proceedings of the 26th International

[17] USA, 129–136. https://doi.org/10.1145/2645710.2645746 Elena Smirnova and Flavian Vasile. 2017. Contextual Sequence Modeling for Recommendation with Recurrent Neural Networks. arXiv preprint arXiv:1706.07684 (2017).

[18] Bartlomiej Twardowski. 2016. Modelling Contextual Information in SessionAware Recommender Systems with Neural Networks. In Proceedings of the 10th ACM Conference on Recommender Systems (RecSys '16). ACM, New York, NY, USA, 273–276. DOI:https://doi.org/10.1145/2959100.2959162

[19] Bartlomiej Twardowski. 2016. Modelling Contextual Information in SessionAware Recommender Systems with Neural Networks. In Proceedings of the 10th ACM Conference on Recommender Systems (RecSys '16). ACM, New York, NY, USA, 273–276. DOI:https://doi.org/10.1145/2959100.2959162

[20] USA, 129–136. https://doi.org/10.1145/2645710.2645746 Elena Smirnova and Flavian Vasile. 2017. Contextual Sequence Modeling for Recommendation with Recurrent Neural Networks. arXiv preprint arXiv:1706.07684(2017)

[21] Kriesel D., "A Brief Introduction to Neural Networks", Publisher: dkriesel.com, 244 pages, 2011.

[22]D. Kim, C. Park, J. Oh, S. Lee, and H. Yu, "Convolutional matrix factorization for document context-aware recommendation," in Proceedings of the 10th ACM Conference on Recommender Systems (RecSys'16),2016, pp. 233–240.

[23] T. Bansal, D. Belanger, and A. McCallum, "Ask the gru: Multi-task learning for deep text recommendations," in Proceedings of the 10th ACM Conference on Recommender Systems (RecSys'16), 2016, pp.

107–114

[24] *Carlos A Gomez-Uribe and Neil Hunt. 2016. The netflix recommender system: Algorithms, business value, and innovation. TMIS 6, 4 (2016), 13.*