N° **Project: BIOINFO 04**

People's Democratic Republic of Algeria

Ministry of Higher Education and Scientific Research

HOUARI BOUMEDIENE University of Science and Technology

Faculty of Computer Science

Master's thesis

Submitted for the degree of **MASTER**

In : IT

Speciality : Bioinformatics

By : M. MOUHOUB Massinissa, M. ZIGADI Sadji Amine

Proposed by : Pr. MEHDI Malika, Pr. HENNI Khadidja

Theme

Development of a therapeutic recommendation system for osteoarthritis of the knee

Mme.	BOULKRINAT Samia	Professor	at USTHB	President
Mme.	KERKAD Amira	Professor	at USTHB	Member

We dedicate this work to all those who believed in us and supported us throughout this journey.

To our parents, for their invaluable love, patience, and daily sacrifices.

To our families, for their presence, encouragement, and kindness.

To our teachers and supervisors, for their dedication and valuable guidance.

To our friends, for their moral support and constant encouragement.

And to all those who, directly or indirectly, contributed to the completion of this project.

Thank you.

Acknowledgements

First and foremost, we thank God.

We would like to express our deep gratitude to our parents and families for their unconditional support, constant encouragement, and silent sacrifices that have been essential throughout this journey. Our warmest and most heartfelt thoughts go especially to our mothers, whose love, patience, and kindness carried us through moments of effort and doubt.

We also sincerely thank our supervisors, Mrs. Mehdi and Mrs. Henni, for their availability, insightful guidance, scientific rigor, and kind support, which greatly contributed to the quality and successful completion of this work.

Finally, our thanks also go to Abdelhak Kaci Aissa and Manel Arfi for their valuable assistance, constructive feedback, and enriching discussions, which have significantly helped us improve our project.

ملخص

تتناول هذه الدراسة الحاجة الأساسية إلى توصيات شخصية للتمارين العلاجية للأشخاص المصابين بالفُصال العظمي في الركبة. تُظهر التقييمات التجريبية الأداء المتفوق لنموذج الشبكات العصبية الرسومية، حيث حقق نموذج GNN درجة RPS بلغت 91.20%، مقارنةً بـ 64.70% لنموذج MF. وتُبرز هذه النتائج الإمكانات الكبيرة لـ GNN في تحسين أنظمة التوصية في المجال الصحي.

الكلمات المفتاحية : أنظمة التوصية الطبية، الشبكات العصبية الرسومية، التصفية التعاونية.

Abstract

This study addresses the essential need for personalized therapeutic exercise recommendations for individuals with knee osteoarthritis. Experimental evaluations demonstrate the superior performance of the GNN model, achieving an RPS score of 91.20%, compared to 64.70% for the MF model. These results highlight the potential of GNNs to enhance health recommendation systems.

Keywords: Health-care Recommender Systems, Graph Neural Networks, Matrix Factorization (MF), Collaborative filtering.

Résumé

Cette étude répond au besoin essentiel de recommandations personnalisées d'exercices thérapeutiques pour les personnes souffrant d'arthrose du genou. Les évaluations expérimentales démontrent la performance supérieure du modèle GNN, avec un score RPS de 91,20%, contre 64,70% pour le modèle MF. Ces résultats soulignent le potentiel des GNN pour améliorer les systèmes de recommandation de santé.

Mots clés : Systèmes de recommandation médicale, Réseaux de neurones graphiques, Filtrage collaboratif.

Contents

Contents

List of Figures

List of Figures

List of Tables

List of Algorithms

General introduction

Gonarthrosis, a common condition, significantly affects the quality of life of many people around the world, particularly the elderly. Characterized by progressive wear and tear of the cartilage covering the knee bones, it causes pain, joint stiffness, and loss of mobility, hindering everyday movements such as walking or climbing stairs. The complexity of the knee joint, composed of several bones, muscles, tendons, and ligaments, requires a thorough understanding of its anatomy and function in order to develop effective therapeutic strategies.

Traditional approaches to managing knee osteoarthritis often involve prescribing physical exercises aimed at strengthening the periarticular muscles, improving mobility, and reducing pain. However, the effectiveness of these standardized exercise programs is often limited due to the specific needs of each patient, which are influenced by their morphology, age, pain level, and biomechanical characteristics. This variability highlights the crucial importance of personalizing exercise recommendations to optimize therapeutic benefits and improve patient adherence to treatment.

Recent technological advances, such as the KneeKG machine, allow for accurate measurement of the dynamic movement of the knee during the walking cycle. These objective, non-invasive, and radiation-free kinematic data provide an accurate picture of joint function, mobility, and limitations. The use of this dynamic data via artificial intelligence techniques, and more specifically recommendation systems, represents a promising avenue for the design of highly personalized therapeutic exercise programs.

Recommendation systems, widely adopted in various sectors such as e-commerce and entertainment, are essential machine learning algorithms that predict, filter, and identify items likely to interest a user. Their evolution has progressed from traditional collaborative filtering methods to more advanced approaches based on deep learning, notably graph neural networks (GNNs). GNNs are distinguished by their ability to model complex relationships within graph-structured data, where each node can have a variable number of connections and unique characteristics, making them particularly well suited for healthcare recommendation applications.

Nomenclature

The present study aims to develop a system capable of identifying the most appropriate exercises for each patient with knee osteoarthritis based on their biomechanical data. To do this, we explore two main approaches: graph neural networks (GNNs) and matrix factorization (MF). The GNN approach is designed to exploit the complex relational structures between patients and exercises in a bipartite graph, while MF serves as a comparative reference to evaluate its effectiveness in capturing latent preferences from interaction data. A major challenge, the cold start problem, is also addressed using a model that projects the characteristics of new patients into the latent space learned by the models. This work thus lays the foundations for optimized therapeutic intervention through exercise personalization.

Chapter 1

Context and Problem Statement

1.1 Introduction

Gonarthrosis, or knee osteoarthritis, is a common condition affecting a large number of individuals worldwide, particularly the elderly [1, 2]. It is characterized by the progressive degeneration of the cartilage covering the bones of the knee joint [3]. This degeneration leads to pain, joint stiffness, reduced mobility, and discomfort during daily activities such as walking or climbing stairs [2].

Osteoarthritis is a widespread disorder that significantly impairs knee function, making an understanding of its anatomy essential for developing effective exercise recommendations. Numerous scientific studies [4] highlight the prevalence and impact of knee osteoarthritis, thus justifying the relevance of this study in establishing a solid knowledge base on the affected joint.

To better understand this condition and its consequences, it is important to examine the structure and functioning of the knee. The knee is a highly complex joint composed of several bones, muscles, tendons, ligaments, and menisci. These components work in unison to enable movements such as flexion, extension, and rotation. When any of these structures are damaged, as in the case of gonarthrosis, it can result in pain and loss of stability.

This chapter therefore presents the anatomical foundations of the knee, the different tissues involved, and their respective roles. It then introduces gonarthrosis and its effects on the joint. Finally, it describes a modern tool called KneeKG, which enables dynamic measurement of knee movements. These data, known as kinematic data, will be highly valuable for the remainder of this work, as they allow for a more accurate analysis of each patient's condition and the proposal of a personalized treatment plan.

1.2 Knee Anatomy

The knee joint, the largest joint in the human body [5], connects the thigh and the lower leg. It encompasses the knee articulation itself. This joint is a complex biomechanical structure and is frequently affected by traumatic injuries and chronic internal disorders [6]. To understand knee-related conditions such as osteoarthritis—commonly referred to as gonarthrosis—it is essential to have a thorough understanding of this intricate anatomical structure [7]. In the following section, we examine the anatomy of the knee by exploring its major components: bones, muscles, tendons, ligaments, and menisci.

1.2.1 Bony Structures

The knee is generally considered to be composed of the femorotibial and femoropatellar joints. The bony part of the knee consists of the femur, tibia, patella, and fibula [7], as illustrated in Figure 1.1. Fractures may affect any of these four bones. Some fractures require immobilization and a prohibition on weight-bearing, others necessitate early rehabilitation, and some cases call for surgical intervention [8].

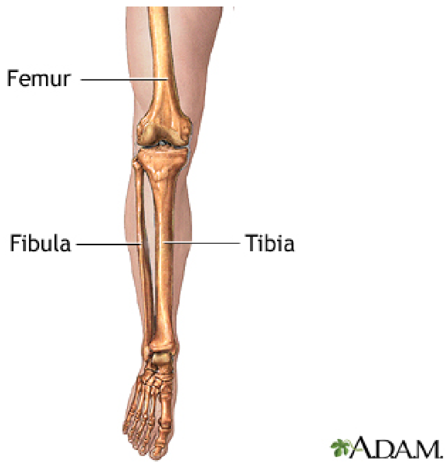


Figure 1.1. Bony structures of the knee [9].

The Femur

One of the largest bones connected to the knee is the femur. It is the longest and strongest bone in the human body and plays a crucial role in posture and balance [10, 11]. The femur articulates proximally with the ilium and distally with the tibia, as shown in Figure 1.1 and Figure 1.2. It serves as an attachment point for major thigh muscles, including the quadriceps femoris and the articularis genus [12].

Its distal end[1] has specific anatomical features critical for knee articulation. The medial and lateral condyles are rounded articular surfaces covered with hyaline cartilage that articulate with the tibial plateaus[2] [5]. Fractures of the femur generally occur only as a result of high-energy trauma, such as car accidents, unless the bone is weakened by conditions like osteoporosis.

The complex anatomy of the distal femur highlights its essential role in the biomechanics of the knee joint and its vulnerability to arthritic changes. The condyles and the femoral trochlea are directly involved in joint articulation and weight-bearing, making their cartilage particularly susceptible to degeneration [13].

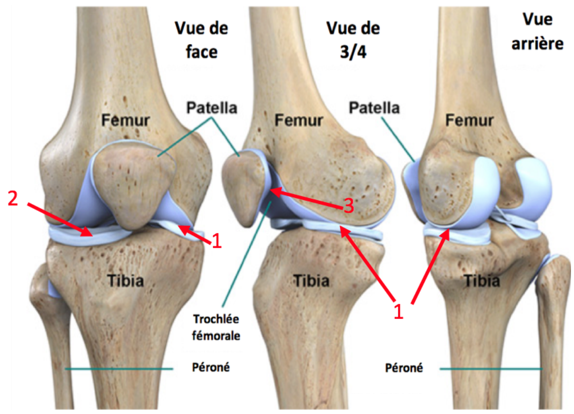


Figure 1.2. Knee bones from different angles [14].

The Tibia

The tibia, also referred to as the shinbone—as illustrated in Figure 1.3 and Figure 1.2—is one of the two longest bones in the leg and serves as the primary weight-bearing bone [15]. It plays a fundamental role in knee joint function [6].

Its proximal end[3] includes the medial and lateral condyles—flat or slightly concave articular surfaces that articulate with the femoral condyles to form the tibiofemoral joint [5]. The tibia is a robust bone, and its widened proximal extremity forms the tibial plateau, which articulates with the femoral condyles to constitute the tibiofemoral joint.

The tibial plateau consists of two distinct surfaces separated by the intercondylar eminences, which serve as attachment points for the cruciate ligaments. The

[1] The distal end of the femur is the lower portion of the thigh bone that articulates with the tibia and patella to form the knee joint.

[2] The tibial plateau is the upper articular surface formed by these condyles.

[3] The proximal end of the tibia, or proximal epiphysis, is the upper portion of the tibia that articulates with the femur and the fibula to form the knee joint.

articular surfaces of the tibia are covered with hyaline cartilage, allowing for smooth articulation with the femur [6].

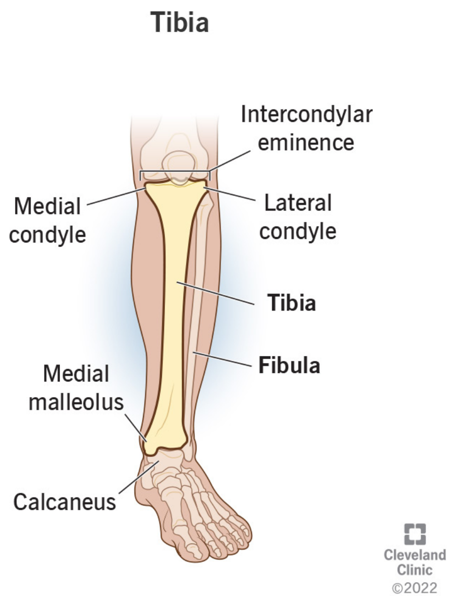


Figure 1.3. Tibia anatomy [16].

The Fibula

The fibula[4] is a long, slender bone located laterally to the tibia and does not bear significant weight. Its proximal end consists of the head, which articulates with the lateral tibial condyle at the proximal tibiofibular joint, and the neck, a narrow region below the head that is a common site for fibular nerve injury [15]. It is characterized by its delicate structure and smaller size, making it the thinnest bone in the human body [17]. Although often overshadowed by the tibia, which plays the main role in bearing body weight, the fibula is essential for lower limb biomechanics. It significantly contributes to ankle joint stability and serves as an attachment site for numerous muscles crucial for foot and ankle movements [5].

The Patella

The patella is a triangular sesamoid bone located at the front of the knee joint, embedded within the quadriceps tendon [5], as shown in Figure 1.5. It has a superior base, an inferior apex, and medial and lateral borders [19]. Its posterior surface is covered with articular cartilage and divided into several facets: a small medial facet,

[4]Also referred to as the *fibula* in English.

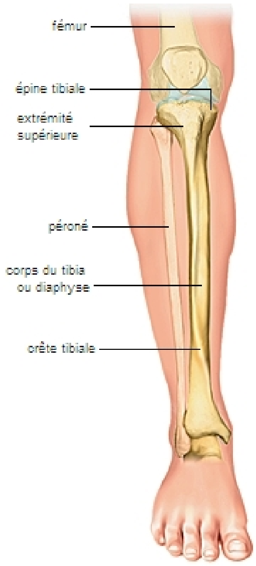


Figure 1.4. The fibula [18].

a larger lateral facet, and an odd facet (inconstant, located at the medial edge). These facets ensure optimal contact with the femoral trochlea during knee flexion-extension movements [6]. The patella protects the anterior aspect of the knee joint [20] and enables smooth movement during knee flexion and extension [21]. Its unique position and function make it a key component of the knee extensor mechanism and a frequent site of arthritic changes affecting the femoropatellar joint.

1.2.2 Muscles

Muscles are soft tissues composed of extensible fibers that contract to produce movement [22]. Knee movements are primarily controlled by two major muscle groups in the thigh: the quadriceps in the front and the hamstrings in the back [23].

The **quadriceps femoris**, located at the front of the thigh (see Figure 1.5), is a powerful muscle group made up of four heads: the rectus femoris, vastus lateralis, vastus intermedius, and vastus medialis [22]. The three vasti originate from the femur, while the rectus femoris arises from the pelvis. The primary function of the quadriceps is knee extension, i.e., straightening the leg. It is also the most important muscle for stabilizing the knee joint. Its role in knee extension and stabilization is essential for weight-bearing activities and exercises relevant for osteoarthritis patients [23].

The **hamstring muscles**, located at the back of the thigh (see Figure 1.6), include the biceps femoris, semitendinosus, and semimembranosus [22]. Except for the short head of the biceps femoris, which originates from the femur, these muscles

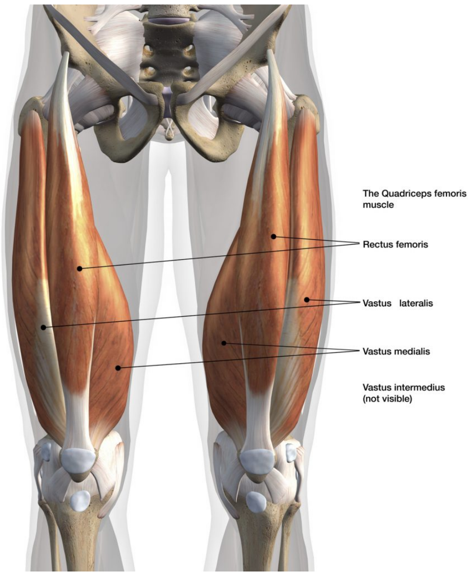


Figure 1.5. The quadriceps muscle [24].

arise from the ischial tuberosity and insert into the proximal parts of the tibia and fibula. Their main function is knee flexion, i.e., bending the leg. The hamstrings act as antagonists to the quadriceps, and balance between these two muscle groups is essential for healthy knee movement [23].

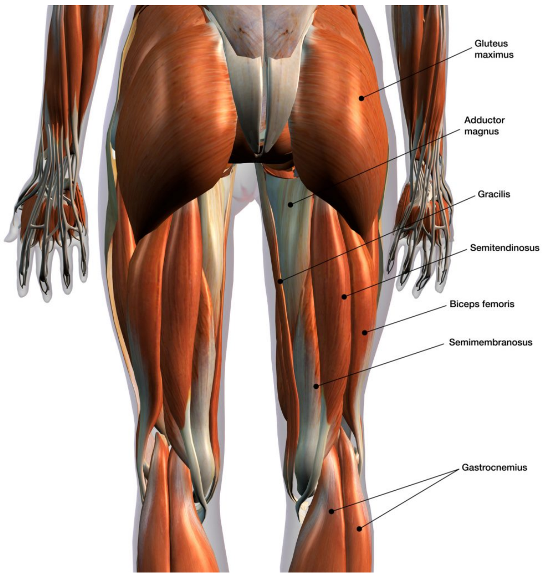


Figure 1.6. The hamstring muscles [24].

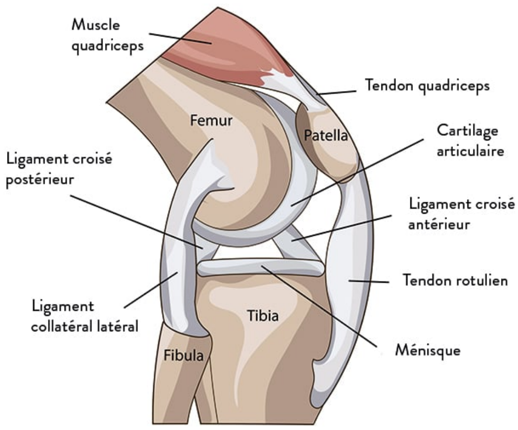


Figure 1.7. Knee tendons [25].

1.2.3 Tendons

Tendons are very rigid fibrous structures that connect muscles to bones, transmitting the force generated by muscle contraction to produce movement [23, 8]. Several major tendons are associated with the knee muscles.

The patellar tendon connects the patella to the tibial tuberosity[5] [23], as shown in Figure 1.8. It constitutes the inferior extension of the quadriceps femoris tendon and plays an essential role in the knee extension mechanism [26]. The patellar tendon is a key component of the extensor apparatus, directly transmitting the force generated by the powerful quadriceps muscles to straighten the leg.

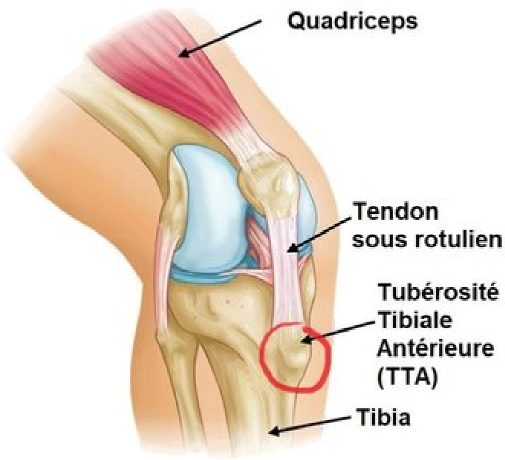


Figure 1.8. The patellar tendon [27].

The quadriceps tendon connects the four quadriceps muscles to the superior part of the patella [23], as illustrated in Figure 1.7. It works in synergy with the

[5]The tibial tuberosity is a bony prominence located on the upper anterior part of the tibia.

patellar tendon to enable knee extension [26]. The quadriceps tendon constitutes the proximal attachment point of the quadriceps muscles to the patella and represents an integral component of the force transmission system required for knee extension.

The hamstring tendons insert around the knee joint, contributing to both flexion and rotation [23], as shown in Figure 1.7. These tendons, attaching to the posterior aspect of the knee, are responsible for leg flexion and also assist in rotational movements.

It is important to note that the tendons surrounding the knee can be affected by osteoarthritis, leading to tendinopathy[6] (e.g., patellar tendinopathy) [28]. This may constitute an additional source of pain, adding to the joint discomfort caused by osteoarthritis [29]. Osteoarthritis may be associated with tendon disorders around the knee, contributing to pain and functional limitations, which must be taken into account when designing exercise recommendations.

1.2.4 Ligaments

Ligaments are strong fibrous bands of connective tissue that link bones together, ensuring joint stability and limiting excessive or abnormal movements. The ligaments of the knee are categorized into collateral ligaments and cruciate ligaments [22].

The collateral ligaments include [22]:

- **The medial collateral ligament (MCL)** — located on the inner (medial) side of the knee, connects the femur to the tibia, as illustrated in Figure 1.7. It stabilizes the medial side of the knee by preventing excessive valgus[7] movement and external rotation.
- **The lateral collateral ligament (LCL)** — located on the outer (lateral) side of the knee, connects the femur to the fibula, as illustrated in Figure 1.7. It stabilizes the lateral side of the knee by preventing excessive varus[8] movement and external rotation.

The cruciate ligaments, located inside the knee joint, cross each other to form an “X” (see Figure 1.7) and control anterior-posterior movements of the knee. They include [22]:

- **The anterior cruciate ligament (ACL)** — located at the front of the knee, prevents anterior displacement of the tibia relative to the femur and posterior rolling of the femur during flexion; it also resists rotational displacement [30].

[6] Tendinopathy is a general term for a painful condition affecting a tendon.
[7] A movement toward the inside.
[8] A movement toward the outside.

- **The posterior cruciate ligament (PCL)** — located at the back of the knee, behind the ACL, prevents posterior displacement of the tibia relative to the femur and excessive anterior movement of the femur on the tibia; it is often considered the primary stabilizer of the knee [23, 31].

In osteoarthritis, the ligaments around the knee may weaken or become overstretched, contributing to joint instability [4]. Ligament injury is considered a potential cause of osteoarthritis [32]. Osteoarthritis may lead to ligamentous alterations, inducing increased laxity and instability, which can further promote disease progression and influence exercise recommendations.

1.2.5 Menisci

The menisci are two crescent-shaped fibrocartilaginous pads located between the femur and tibia in each knee [23]. They act as shock-absorbing cushions, increasing the congruence between the tibial articular surface and the rounded femoral condyles [33].

The **medial meniscus**, situated on the inner side of the knee [23], as shown in Figure 1.9, has a “C” shape and is less mobile. It covers 51% to 74% of the medial articular surface [34]. Due to its shape and attachments, the medial meniscus plays a crucial role in load transmission and stability on the medial side of the knee, and its connection to the MCL makes it susceptible to injury in case of MCL trauma.

The **lateral meniscus**, located on the outer side of the knee, as illustrated in Figure 1.9, is more circular and more mobile than the medial meniscus [23]. It covers a larger portion of the tibial articular surface (75% to 93%) [34]. Thanks to its greater mobility, the lateral meniscus contributes to load distribution and stability on the lateral side of the knee, and its unique attachments provide it with a greater range of motion compared to the medial meniscus.

Meniscal tears are common, especially with aging and in individuals suffering from knee arthritis [23]. Degenerative tears may occur without specific trauma and are frequently associated with osteoarthritis [36]. Meniscal lesions can lead to cartilage wear and increase the risk of developing osteoarthritis or accelerate its progression [37].

1.2.6 Knee Function

The knee joint primarily functions as a synovial hinge joint, allowing flexion and extension [22]. It also permits a slight degree of medial and lateral rotation, especially when the knee is flexed [23]. The muscles, mainly the quadriceps and hamstrings,

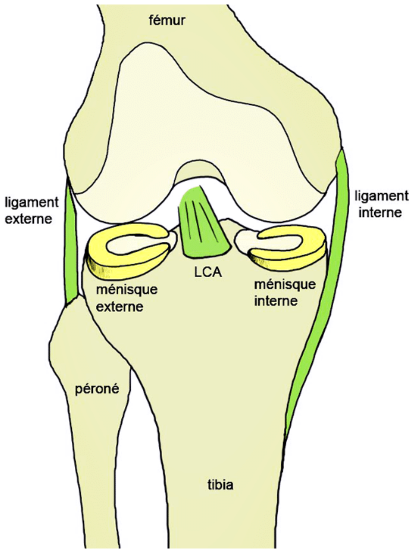


Figure 1.9. The knee menisci [35].

generate the primary movements of flexion and extension. Tendons transmit these muscular forces to the bones (femur and tibia) via the patella. The ligaments, both collateral and cruciate, play a crucial role in maintaining stability during these movements by preventing excessive motion in various planes. The menisci serve a vital function in shock absorption and load distribution when the knee moves and bears weight, thereby protecting the articular cartilage. Osteoarthritis disrupts the coordination of knee structures, reducing its mobility and the ability to perform complex movements.

1.3 Diagnosis of Osteoarthritis and Knee Osteoarthritis

Osteoarthritis is the most common form of arthritis worldwide, affecting a large portion of the population and representing a major cause of chronic pain and disability [38, 39]. Knee osteoarthritis, the most frequent form, specifically affects the knee joint, which is the most commonly involved joint [40, 41]. Its prevalence increases with age, impacting a growing proportion of elderly individuals [42], which poses a significant challenge for healthcare systems. Approximately 20% of the global population is affected, with regional variations [43]. Beyond physical pain, osteoarthritis also causes psychological suffering and socio-economic consequences [44].

Clinically, knee osteoarthritis can be diagnosed using two main approaches. The first relies on radiological (X-ray, MRI) or arthroscopic examinations, which evaluate the structural integrity of the knee. However, these examinations are static and do not provide information on the dynamic function of the joint [45]. To overcome this limitation, the KneeKG device allows dynamic analysis of the knee by extracting kinematic components during the patient's gait cycle.

1.4 KneeKG: The Device Used for Knee Diagnosis

KneeKG is a technology developed by the company *Emovi* (Emotion • Movement • Vision [46]), representing the first method to accurately measure the dynamic 3D alignment of the knee. This system provides objective data helping surgeons make informed decisions to correct alignment and restore the knee's three-dimensional function [46].

The exoskeleton shown in Figure 1.10 is a fixation device designed to eliminate artifacts related to skin and muscle movements. It is placed on the patient's lower limb. Through an optical system, the 3D kinematics of the knee is captured while the patient walks on a treadmill at a comfortable speed chosen by the patient, in a dynamic and weight-bearing context. Data are recorded along three planes of motion.

The KneeKG system can be used in clinics, hospitals, or imaging centers. After approximately five minutes of warm-up on the treadmill, the clinician fixes the KneeKG components on the patient's leg to track joint movements. After calibration, two walking trials of 45 seconds each are recorded, allowing observation of multiple gait cycles. The collected data are then objectively analyzed, providing reliable information on knee function.

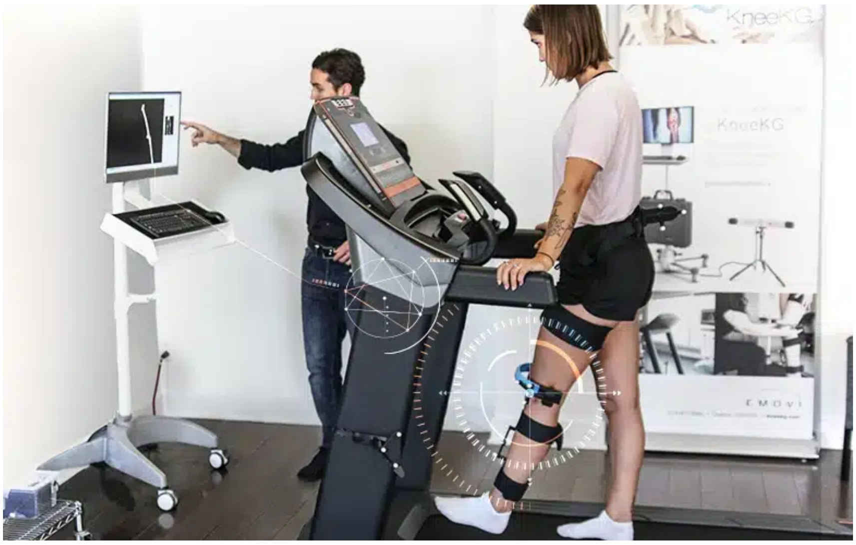


Figure 1.10. KneeKG exoskeleton fixed on the patient's leg [47].

1.5 Kinematic Data

Kinematic data of the knee joint correspond to dynamic movement measurements obtained during gait cycle analysis, as illustrated in Figure 1.11, thanks to the KneeKG system. These data are increasingly used to quantify knee function [48], understand pathological alterations [49], and evaluate disease progression and its impact on gait [50, 51, 52]. They also offer opportunities for diagnosis [53], classification [54], and treatment of musculoskeletal knee pathologies, particularly osteoarthritis [55].

Several studies [51, 54, 56, 57] have used these data to differentiate knees affected by symptomatic osteoarthritis from asymptomatic subjects [58] and classify levels of disease severity.

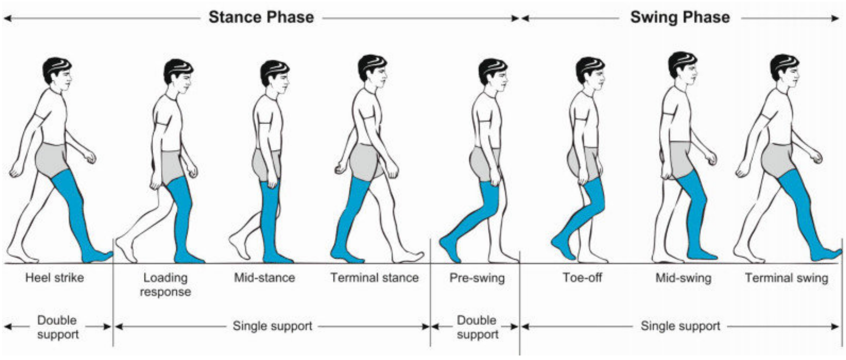


Figure 1.11. Standard human gait cycle [59].

Kinematic data include ground reaction forces (GRF), decomposed into three orthogonal components F_x, F_y, F_z (antero-posterior, medio-lateral, and vertical), as illustrated in Figure 1.12, as well as relative tibia and femur displacements along three axes. These measurements characterize knee movements (flexion/extension, abduction/adduction, internal/external rotation) in dynamic conditions [56].

These data present several major advantages:

- *Reflect the functional state of the knee*: unlike static images, they reveal the real dynamics of the moving knee, its mobility, limitations or compensations, offering a precise view of joint function.
- *Objective, non-invasive, and radiation-free measurements*: collected by external sensors (see Figure 1.13), they avoid subjective interpretation biases and radiation risks.
- *Allow personalized care*: healthcare professionals can monitor disease progression, adapt treatments, and propose individualized care pathways based on

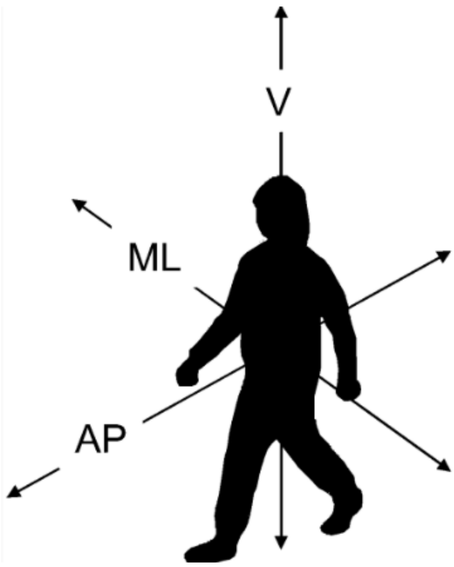


Figure 1.12. The three components of kinematic data [60].

the knee's dynamic data.

In summary, these data allow for more precise, safe, and tailored management of patients suffering from knee osteoarthritis [61].

Figure 1.13. The sensors of the KneeKG machine [62].

1.6 Problem Statement

Knee osteoarthritis is a chronic disease affecting the joint, causing pain, stiffness, and loss of mobility, significantly reducing quality of life. Physical exercises are often prescribed to strengthen periarticular muscles, improve mobility, and reduce pain. However, each patient has specific needs related to their morphology, age, pain level, and biomechanical characteristics. Providing the same exercise program to all may therefore be ineffective.

Personalization of exercises based on individual data is essential. Today, thanks to technologies such as KneeKG, it is possible to accurately measure dynamic knee movement. These data can be exploited by artificial intelligence techniques, detailed in the following chapters, to design a personalized exercise recommendation system. This work aims to develop a system capable of identifying the most suitable exercises for each patient based on their biomechanical data. Inspired by recommendation systems used in other fields, this approach seeks to optimize therapeutic benefits and adherence.

1.7 Conclusion

This chapter presented the anatomy and function of the knee, as well as knee osteoarthritis, a disease damaging the joint causing pain and motor difficulties. We introduced the KneeKG technology, which finely analyzes joint movement to better understand patients' specific needs. Finally, we highlighted the necessity to adapt physical exercises, using dynamic kinematic data, to improve the care of people affected by osteoarthritis.

The next chapter will be devoted to a review of existing work on recommendation systems in rehabilitation, relevant artificial intelligence algorithms, and methodological bases to design a system adapted to this medical context.

Chapter 2

Recommendation Systems

2.1 Introduction

In today's digital landscape, individuals frequently face an overabundance of information and options across various domains, ranging from e-commerce platforms to streaming services and online resources. This information overload complicates users' tasks when it comes to identifying content or products that match their specific needs and preferences. Recommendation systems have thus established themselves as an essential class of machine learning algorithms designed to address this challenge by predicting, filtering, and identifying items likely to interest a user among a vast array of possible choices [63].

These systems enable the personalization of exploration within information spaces by prioritizing relevant items for a given user, thereby reducing search effort while enhancing the overall interaction experience. The evolution of recommendation systems has witnessed significant progress, moving from classical collaborative filtering methods to more advanced approaches based on deep learning. Among these advances, Graph Neural Networks (GNNs) stand out for their ability to model complex relationships within data, representing a paradigm shift in how recommendations are generated.

2.2 General Context of Recommendation Systems

2.2.1 Definition of a Recommendation System

A recommendation system, or RS, aims to generate recommendations for a set of users. These recommendations can be in the form of products such as books, movies, songs, etc. (Often the terms used are *users* to designate the users, and *items* to designate the objects to recommend) that may interest the user. The creation of a recommendation system depends on the domain and the particular characteristics of the available data [64]. The recommendation system uses users' historical data in order to generate recommendations, so this data plays an important role [65].

Ricci *et al.* [66] defined a recommendation system as follows:

> "Recommendation systems (RS) are software and technical tools that provide suggestions of useful items to a user [67, 68, 69]. The suggestions concern various decision-making processes, such as which items to buy, which music to listen to, or which online news to read."

Recommendation systems have become widespread in many fields: e-commerce (Amazon), entertainment (Netflix, Spotify), social networks (YouTube, Facebook), education (MOOCs), and health (treatment recommendations, personalized monitoring).

2.2.2 Classification of Recommendation Approaches

There are several approaches[1] to recommendation systems. Among the most well-known are content-based recommendation systems, collaborative systems, as well as hybrid systems that combine multiple approaches. Moreover, there are also more specific recommendation systems tailored to particular application domains, as illustrated in Figure 2.1.

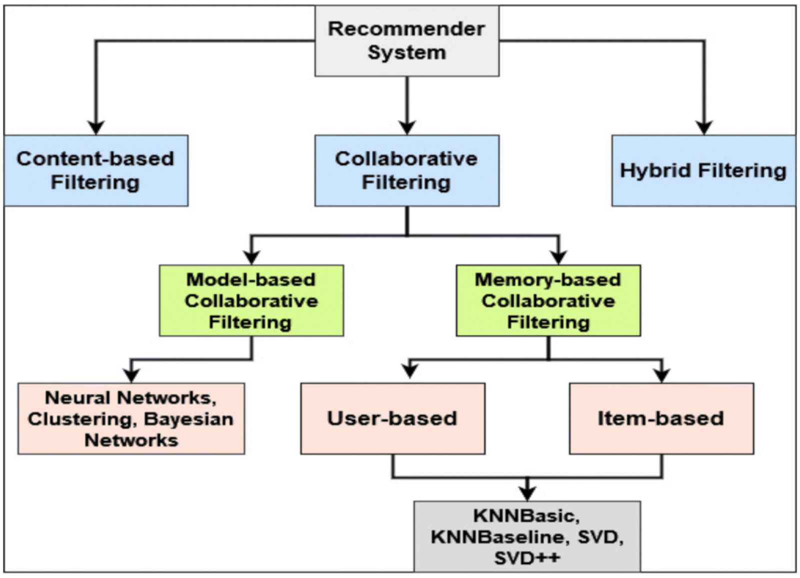


Figure 2.1. Recommendation system approaches [71].

[1] A "recommendation approach" is a model for implementing a recommendation [70].

Content-Based Approach

Content-Based Filtering (CBF) is one of the most common approaches in recommendation systems [66]. It relies on analyzing a user's past interactions to model their preferences — a process called user modeling — by inferring interests from items already viewed or liked [70]. The key principle is that users tend to like content similar to what they have already liked, as illustrated in Figure 2.2 [72]. This approach allows for fine personalization without depending on other users' preferences [70]. However, it has some limitations, such as low serendipity [2] and a risk of overspecialization, where recommendations become too similar to already consumed content [66]. Moreover, it usually does not consider the quality or popularity of items, which can harm the diversity of suggestions. Despite these limitations, CBF remains widely used, especially in fields such as scientific article recommendation [70].

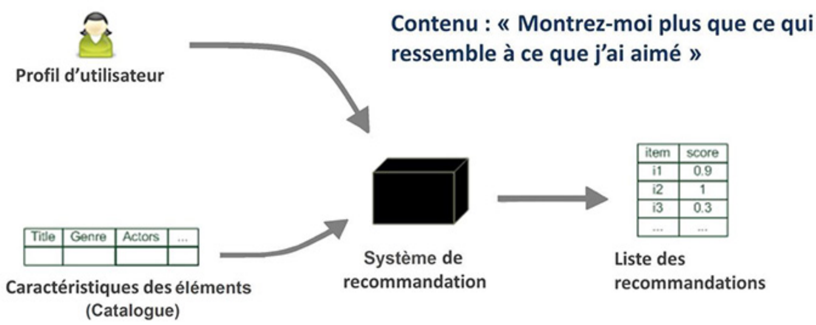


Figure 2.2. Content-based approach [73].

Collaborative Approach

Collaborative Filtering (CF) is based on the idea that users generally like items that others with similar tastes have also liked, as illustrated in Figure 2.3. Two users are considered similar when they give close ratings to the same items. Once these profiles are detected, items liked by one user can be recommended to the other [70, 72].

Compared to Content-Based Filtering (CBF), CF offers several advantages [70]:

- (i) It does not depend on item features [74].
- (ii) It exploits real subjective judgments made by users [75].

[2]The ability of the system to recommend relevant but unexpected items.

- (iii) It fosters more surprising and personalized recommendations, thanks to similarities between users [74].

However, this approach is not without limitations. It is sensitive to the *cold-start* problem[3], and may fail for so-called *gray sheep* users[4], whose tastes are not shared by any clear segment [72, 79].

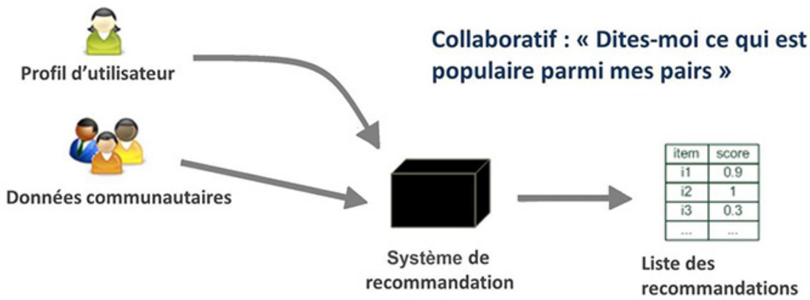


Figure 2.3. Collaborative approach [73].

Hybrid Approach

To combine the advantages of content-based filtering and collaborative filtering, several hybrid approaches have been developed, as illustrated in Figure 2.4. A simple method consists of letting each technique generate its recommendations, then merging the results to produce a final list [80]. Hybrid recommender systems (Hybrid RS) thus combine multiple strategies to strengthen their advantages and compensate for their weaknesses.

One of the earliest examples is *Fab*, a meta web recommendation system combining CF (to find users with similar tastes) and CBF (to identify similar content) [81].

For instance, the "companies you may want to follow" feature on LinkedIn combines both approaches: CF evaluates similarity with companies already followed, while CBF checks that the sector or location matches the user's interests [82].

[3] Cold-start refers to the inability to make relevant recommendations in the absence of sufficient data on new users or items [76, 77].

[4] Users with atypical preferences that do not correspond to any identifiable group [78].

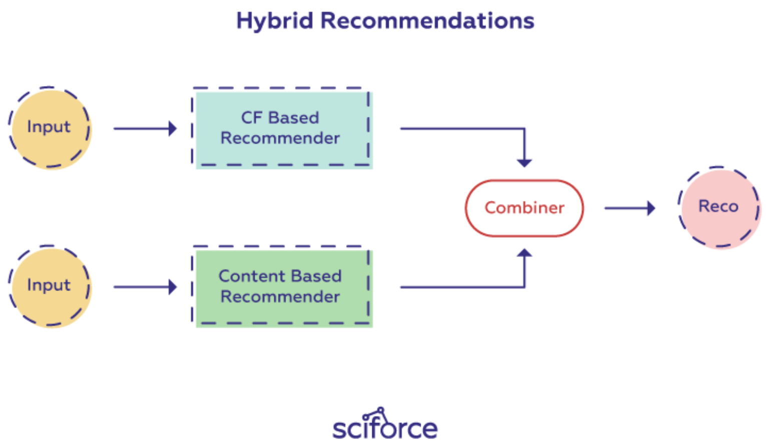


Figure 2.4. Hybrid approach [83].

2.3 Application of Recommender Systems in Healthcare

Healthcare Recommender Systems (HRS) leverage the recommendation system approaches previously mentioned, such as collaborative filtering (CF), content-based filtering (CBF), and hybrid methods, to propose personalized treatments, medications, or care plans [84]. These systems rely on a variety of clinical data — medical history, symptoms, diagnoses, ongoing prescriptions, and even test results — to identify patients with similar profiles and recommend therapeutic options that have proven effective in analogous cases [85].

An example of an HRS is IBM Watson [86, 87], which analyzes large sets of clinical data and suggests relevant treatments using machine learning methods. According to IBM, 81% of healthcare executives familiar with Watson acknowledge its positive impact on their decision-making, highlighting the growing role of data analytics in medical practice [72, 79].

Beyond traditional techniques, HRS also incorporate advanced algorithms such as ant colony optimization [81], clustering, decision trees, logistic regression, support vector machines (SVM), as well as more recent approaches like matrix factorization, medical ontologies, semantic reasoning, and natural language processing (NLP) [84, 85, 88]. These techniques are crucial for representing the complexity of medical data and modeling patients' specific preferences or needs.

Among these methods, knowledge-based approaches[5] play a central role. They

[5]Knowledge-Based Recommender Systems rely on explicit rules and prior knowledge about users and items to recommend. Unlike collaborative or content-based approaches, they do not require

allow the integration of explicit information about patients, such as allergies or drug interactions, into the recommendation process [89]. According to Sadasivam et al. [88], these approaches also help overcome the cold-start problem — one of the biggest challenges in recommender systems [76, 90] — by generating suggestions even in the absence of rating history or similar data.

Thus, the deployment and integration of intelligent algorithms in healthcare systems not only improve the relevance of recommendations but also support healthcare professionals in complex decision-making by providing objective assistance based on clinical and statistical evidence [91].

2.4 Collaborative Recommender System

At its core, a collaborative recommender system is designed to predict a user's interest in new items based on the collective preferences of a community of users [92, 93]. The fundamental components of such a system include a set of users, a set of items, and a mechanism to represent users' preferences for these items, often called a utility function [93, 94]. This utility function can take various forms, such as explicit ratings on a numerical scale or implicit indicators of preference, like purchase history or viewing time. Typically, these preferences are organized into a *user-item* matrix as illustrated in Figure 2.5, where each row represents a user, each column represents an item, and the entries indicate the user's preference for that item [95]. An important characteristic of this matrix in real-world applications is its sparsity, meaning most users have only interacted with a small subset of the total available items [96]. This sparsity presents a major challenge for collaborative filtering algorithms, as they must infer missing preferences based on a limited amount of observed data.

Collaborative filtering differs from other recommender system approaches by its primary reliance on user-item interactions [98]. Content-based systems, on the other hand, focus on the attributes or features of items, recommending items similar to those a user has liked in the past [99]. This approach requires detailed information about the items themselves to determine similarity [100]. Hybrid systems aim to combine the strengths of both collaborative and content-based methods, often resulting in improved accuracy and coverage of recommendations [93].

Within the domain of collaborative filtering, several key problem formulations exist [93]. User-based recommendation involves identifying users who have prefer-

interaction histories. Instead, they use explicit constraints (e.g., "a diabetic patient should not receive drug X") to generate precise recommendations, often in critical contexts such as healthcare or expert systems

Figure 2.5. The user-item matrix [97].

ence patterns similar to a target user, then recommending items that these similar users have liked but the target user has not yet encountered [100], as illustrated in Figure 2.6. Item-based recommendation, conversely, focuses on identifying items similar to those the target user has liked in the past and recommending those similar items [93], as illustrated in Figure 2.6.

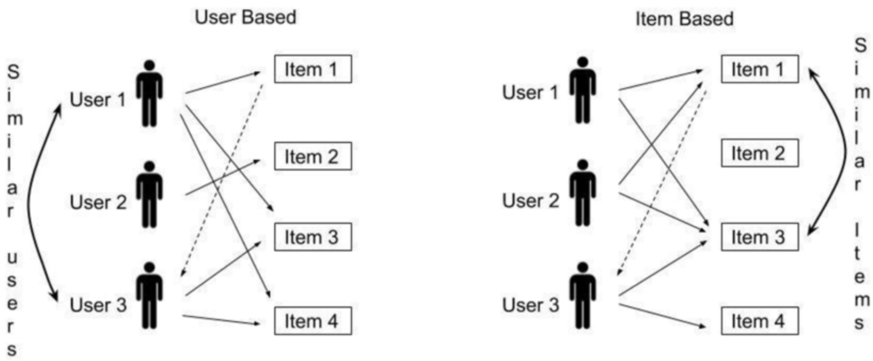


Figure 2.6. User-based recommendation and item-based recommendation [101].

Another formulation consists of predicting individual ratings, where the system aims to estimate the utility or specific rating that a user would give to an item they have not yet rated [93]. Finally, many systems aim to generate *top-N* recommendations, providing a ranked list of the N items that a user is most likely to find interesting [102]. In the context of exercise recommendations for patients with OA (Osteoarthritis), user-based approaches (finding similar patients and their preferred exercises) and item-based approaches (finding exercises similar to those a patient has liked) could be relevant [100], and this will be the focus of our design.

2.4.1 Types of collaborative filtering approaches

Collaborative filtering techniques can be broadly classified into three categories: memory-based methods, model-based methods, and hybrid approaches [103].

We present below these methods and some of the techniques they use.

Memory-based methods

These approaches directly exploit the entire set of user-item interaction data to generate recommendations. They rely on identifying users or items similar to the target user or item, then using the preferences of these neighbors to predict the utility of items for the target [103].

- **User-based collaborative filtering** is a memory-based technique that recommends items to a target user based on the preferences of other users who have exhibited similar behavior [96]. The algorithm identifies users whose tastes and preferences are close to those of the target user, then predicts the utility of an item based on the ratings given to that item by these similar users [93] as illustrated in Figure 2.7. Similarity between users is generally calculated by comparing the rows of the user-item matrix using measures such as cosine similarity or Pearson correlation coefficient [96]. In our case, kinematic data can be used to identify similarity between users.

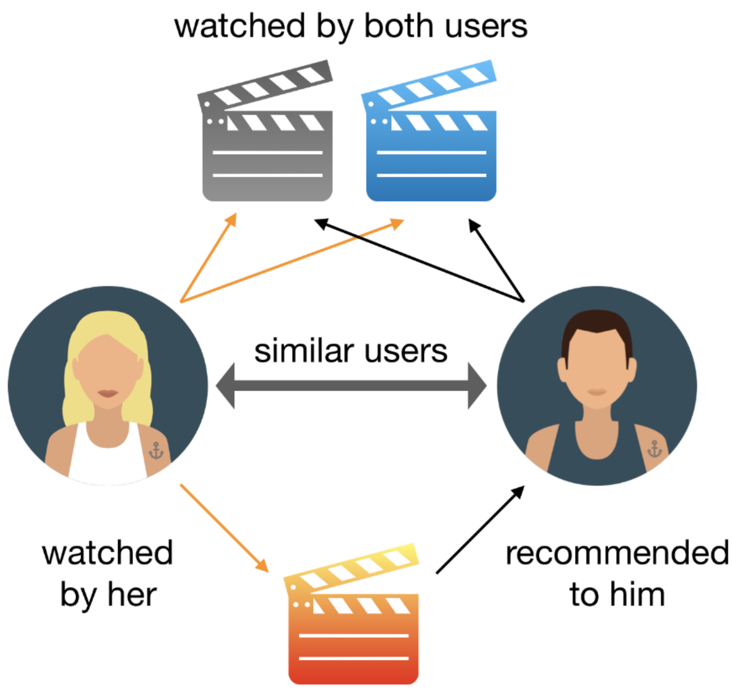


Figure 2.7. User-based collaborative filtering

- **Item-based collaborative filtering** is another memory-based technique that recommends new items to a target user based on how this user has interacted with similar items [96]. The algorithm identifies items similar to those the user has liked in the past, then predicts the utility of a new item based on the degree of appreciation shown towards similar items [96] as illustrated in Figure 2.8. Similarity between items is determined by analyzing the users who have interacted with them, and is generally calculated by comparing the columns of the user-item matrix [93]. In our case, this approach would consist of suggesting exercises presenting characteristics similar to those the patient has already practiced or rated positively, taking into account criteria such as intensity, exercise type, or the body parts involved [104].

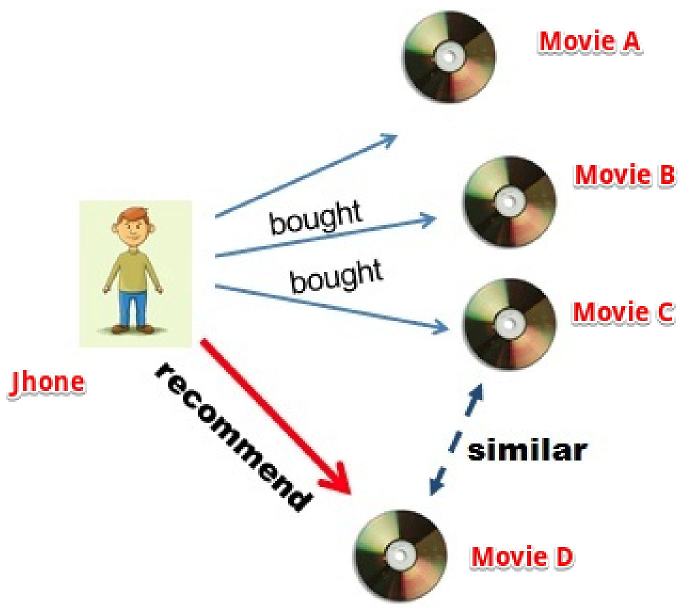


Figure 2.8. Item-based collaborative filtering [105].

Model-Based Methods

These approaches use user-item interaction data to learn a model, which is then used to make predictions [93]. The goal is to capture the underlying patterns and relationships in the data [106].

- **Matrix factorization** is a popular technique that consists of decomposing the user-item matrix into the product of two lower-dimensional matrices representing the latent factors of users and items [93]. Methods such as singular value decomposition (SVD) and non-negative matrix factorization (NMF) are commonly used [107] as shown in Figure 2.9. These latent factors can represent unobserved attributes that influence user preferences or item characteristics [108]. In our case, it can be used to identify latent preferences for certain types of exercises or specific benefits related to certain therapeutic routines.

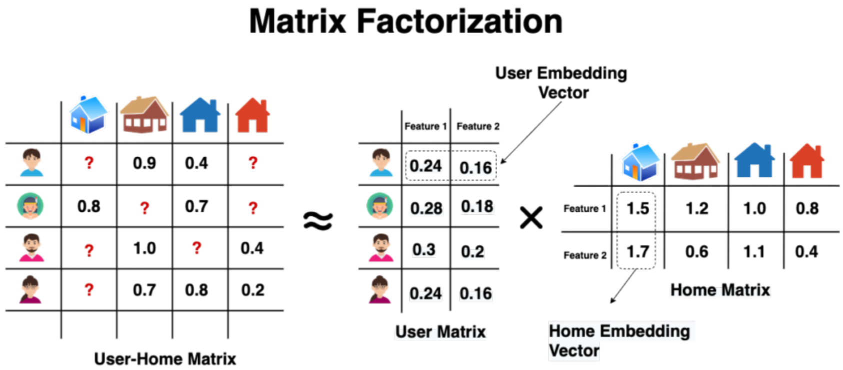


Figure 2.9. Matrix factorization [109].

- **Clustering techniques** consist of grouping users or items into clusters[6] based on their similarity [110], as illustrated in Figure 2.10. Once clusters are formed, predictions for a user can be established based on the preferences of other members in their cluster [111]. In our case, clustering could allow grouping patients who exhibit similar behaviors or responses to certain exercises, thus facilitating the recommendation of exercises within the cluster to which a new patient is assigned.

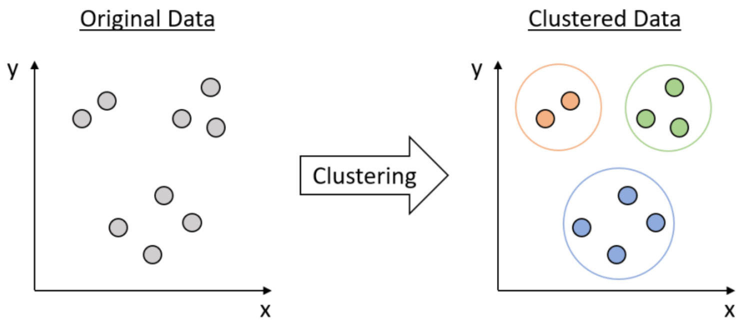


Figure 2.10. Clustering [112].

- **Deep neural networks**, which are illustrated in Figure 2.11, have been introduced for collaborative filtering in order to replace the linear dot product of

[6] A cluster refers to a group of elements (e.g., users or items) that share similar characteristics according to a certain distance or similarity metric.

matrix factorization with nonlinear architectures capable of modeling complex interactions between users and items [113]. In this framework, variants such as GMF, MLP, or NeuMF use multi-layer perceptrons to directly learn the user–item interaction function from implicit data [114].

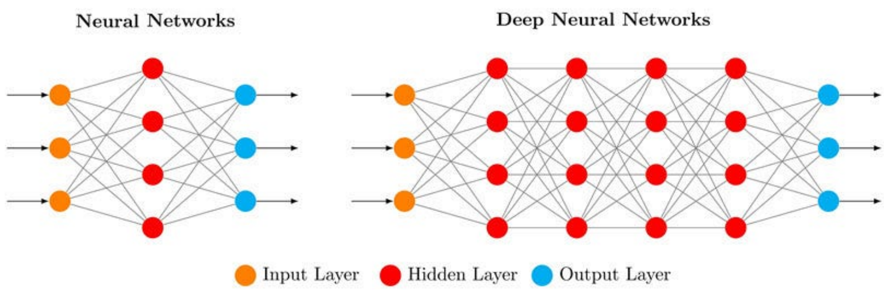


Figure 2.11. Deep neural networks [115].

More recently, Graph Neural Networks (GNN), see Figure 2.12, have been applied to collaborative recommender systems: by considering the user–item information as a graph, GNNs aggregate messages from neighbors to capture structural and relational dependencies. Studies show that these models, thanks to their message-passing mechanism, achieve significant performance gains over traditional approaches on standard datasets [116].

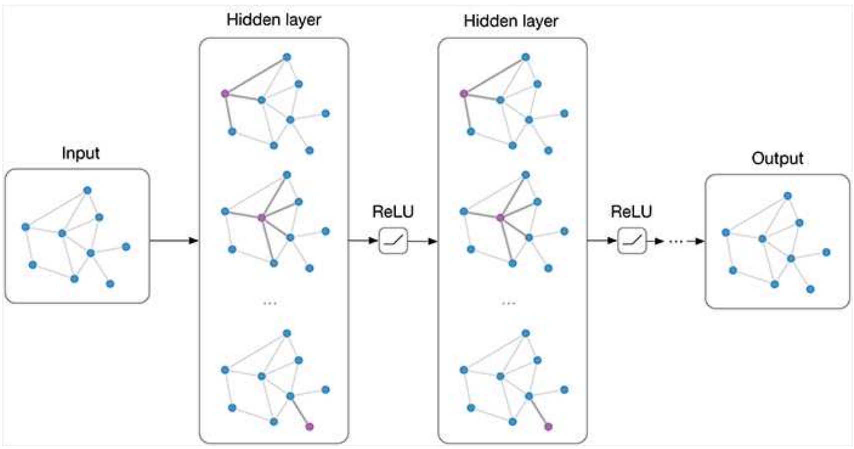


Figure 2.12. Graph Neural Networks (GNN) [117].

2.4.2 Advantages and Limitations of Collaborative Filtering

Collaborative filtering is one of the most proven and widely adopted approaches in the field of recommender systems, due to its numerous functional and methodological advantages [118]. One of its main strengths lies in its ability to generate unexpected or non-trivial recommendations, thus allowing users to discover new interests not explicitly expressed in their previous interactions. Unlike content-based methods, which tend to suggest items similar to those already liked, collaborative filtering exploits the preferences of users sharing analogous behaviors, thereby facilitating the recommendation of items popular among similar profiles, even in the absence of common characteristics [119].

A substantial advantage of this approach is that it does not rely on explicit knowledge of item attributes or content, making it particularly suitable for contexts where feature extraction is imprecise, costly, or subjectively biased [100]. By relying exclusively on actual user interactions, collaborative filtering reflects individual preferences more faithfully than manually produced descriptions [120]. It also stands out for its ability to identify latent or niche preferences within a user community, often imperceptible to other techniques.

Par ailleurs, la performance de ces systèmes tend à s'améliorer avec l'accroissement du volume de données collectées, rendant les recommandations progressivement plus personnalisées et précises au fil du temps. Cette approche présente aussi une flexibilité remarquable, en identifiant des préférences croisées entre genres ou catégories, tout en restant sensible aux évolutions dynamiques des goûts des utilisateurs. Enfin, le filtrage collaboratif offre un rapport coût-efficacité favorable, en réduisant la nécessité d'une annotation manuelle des données, tout en assurant la diversité et la pertinence des recommandations, ce qui en fait une solution particulièrement adaptée à notre cas [120].

2.5 Introduction to Graph Neural Networks

The evolution of recommender systems has seen progression through various techniques, ranging from basic collaborative filtering methods to more sophisticated deep learning-based approaches. Recently, Graph Neural Network (GNN) techniques have emerged as a powerful tool for capturing complex relationships within data, and they have been widely used in recommender systems [121], because most of the information in recommender systems essentially has a graph structure [122, 123], and GNNs have superiority in learning graph representations, thus bringing a paradigm shift in how recommendations are generated [121]. At the level of academic research, numerous studies attest that models based on GNNs outperform previous approaches and establish new state-of-the-art results on public datasets [124, 125].

2.5.1 Definition of GNNs

Graph Neural Networks (GNNs) are a class of deep learning algorithms specifically designed to handle data that can be represented as graphs [126]. A graph is a data structure composed of nodes [7] and edges [8] connecting these nodes, representing entities and their relationships, respectively [126], as illustrated in Figure 2.13.

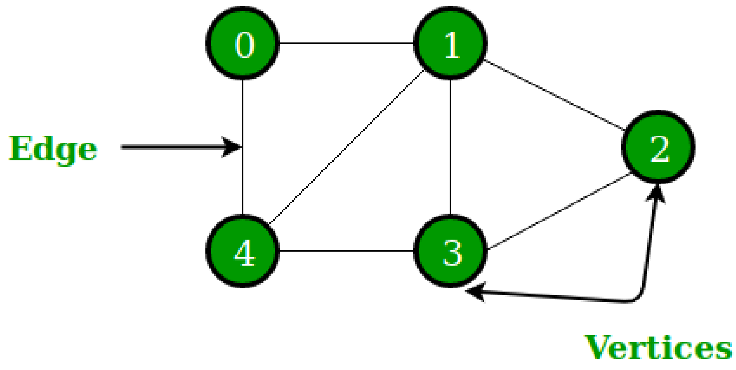
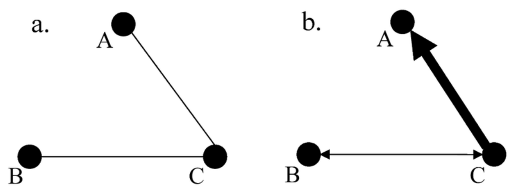


Figure 2.13. Example of a graph structure [127].

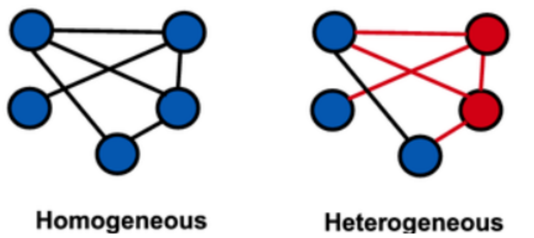
Generally, graphs can be classified as follows:

- **Directed / Undirected Graph**: A directed graph is a graph in which all edges are oriented from one node to another. An undirected graph can be considered a special case of a directed graph, where each link between two nodes is represented by a pair of edges in opposite directions, see Figure 2.14a.
- **Homogeneous / Heterogeneous Graph**: A homogeneous graph consists of a single type of nodes and edges, whereas a heterogeneous graph contains multiple types of nodes or edges, see Figure 2.14b.
- **Hypergraph**: A hypergraph is a generalization of a graph in which an edge (called a hyperedge) can connect any number of nodes, see Figure 2.14c.

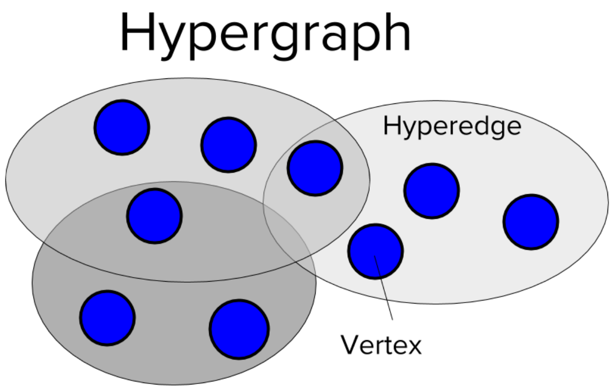
[7] *Vertices* in English.
[8] *Edges* in English.



(a) Directed / Undirected Graph [128].



(b) Homogeneous / Heterogeneous Graph [129].



(c) Hypergraph [130].

Figure 2.14. Graphical representation of graph types.

GNNs are particularly well-suited for applications where the relationships between data points are as important as the points themselves, such as social networks, knowledge graphs, recommender systems, and bioinformatics [131], see Figure 2.15. Unlike traditional neural networks, which operate on grid- or sequence-type data, GNNs can handle the complexity and heterogeneity of graph-structured data, where each node may have a variable number of connections and its own features [132].

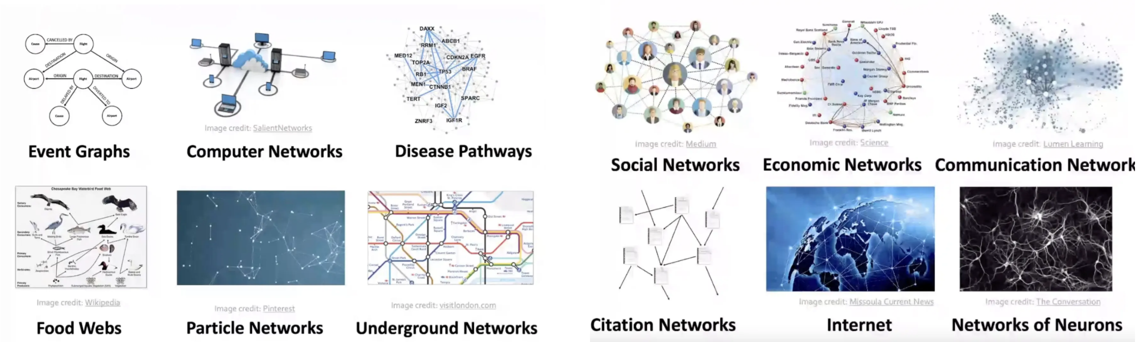


Figure 2.15. Example of graph usage [133].

The development of GNNs is inspired by deep learning algorithms such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), but they are adapted to the non-Euclidean[9] nature of graph data [134]. The central idea of GNNs is to learn representations (or *embeddings*) for nodes, edges, or the entire graph by taking into account both node features and graph topology [121].

Embeddings are continuous vector representations that translate complex entities (such as words, users, or nodes) into vectors usable by machine learning algorithms, as illustrated in Figure 2.16. They capture semantic[10] or structural[11] similarity between these entities by placing them meaningfully in a vector space. Embeddings are used in various domains such as language processing, recommendation, or graphs; they play a crucial role in transforming data into learnable and exploitable features [135, 136].

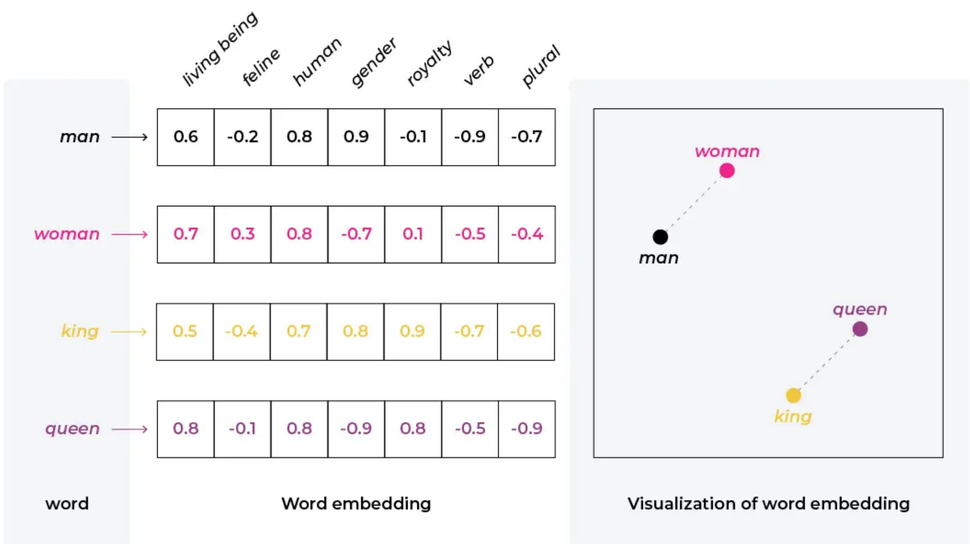


Figure 2.16. Example of word embeddings illustrating semantic relationships between terms [137].

The generation of embeddings in GNNs is done through a message-passing pro-

[9]Data are called Euclidean when elements are organized in a regular space, such as an image, text, or table. In contrast, graph data have no regular structure. The relationships between elements (nodes) can be arbitrary.

[10]Semantic similarity means that two entities have similar meanings or roles, even if they appear different. For example, two users could be considered semantically similar if they like the same movie genres or have comparable tastes, even if they do not know each other.

[11]Structural similarity concerns the position or role of entities in a structure such as a graph. For example, in a social network represented as a graph, two users could be structurally similar if they occupy similar positions in the network, even if they are not directly connected or do not share the same interests.

cess, where nodes exchange information with their neighbors iteratively, aggregating and transforming these messages to update their representations. The ability of GNNs to capture these complex relationships and dependencies explains their growing popularity in various machine learning tasks [121].

2.5.2 Architecture and Operation of GNNs

Similar to traditional deep neural networks, the architecture of a GNN generally relies on multiple fundamental layers specifically designed to effectively process relational data, as illustrated in Figure 2.17.

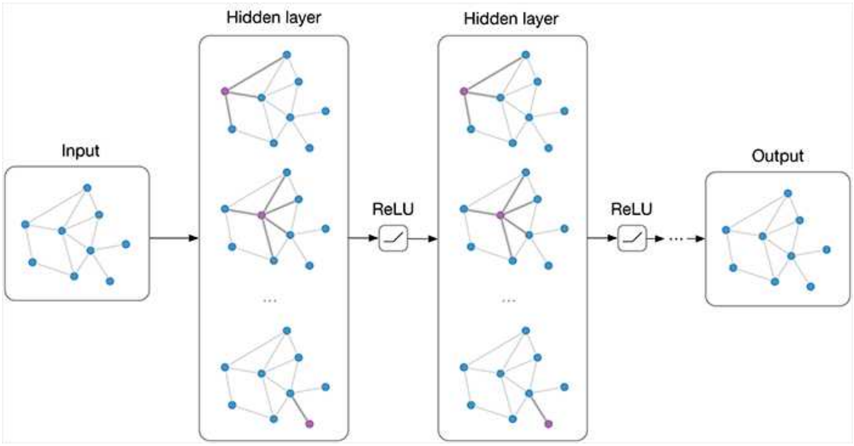


Figure 2.17. Illustration of a GNN architecture with two layers [117].

One of the essential components is the message propagation layer, in which each node aggregates information from its immediate neighbors [138]. This process involves computing neural messages between connected nodes, taking into account their current embeddings and the features of the edges connecting them [139]. For each node, these messages are then aggregated using an aggregation function, such as the mean-pooling operation [140, 141], the attention mechanism [142], etc. The aggregated message is combined with the node's previous embedding via an update function, often nonlinear, to produce a new embedding [139]. This mechanism is iterative and can be applied across multiple layers, thus allowing information to propagate through the graph and to learn from increasingly distant neighbors [143]. The number of layers defines the distance to which information is propagated; a node in a two-layer GNN model will receive information from its immediate neighbors and from the neighbors of its neighbors.

As illustrated in Figure 2.18, node 1 in the image receives the embeddings of its immediate neighbors (here nodes 2 and 3). These embeddings are then aggregated using an aggregation function to synthesize local contextual information. This step is shown in the *AGGREGATE* section of the image. Once aggregation is complete, the node updates its own embedding by combining its current embedding with the aggregated information via an update function, as shown in the *UPDATE* section. This mechanism is iterative and can be applied across multiple layers to allow each node to capture increasingly distant information in the graph.

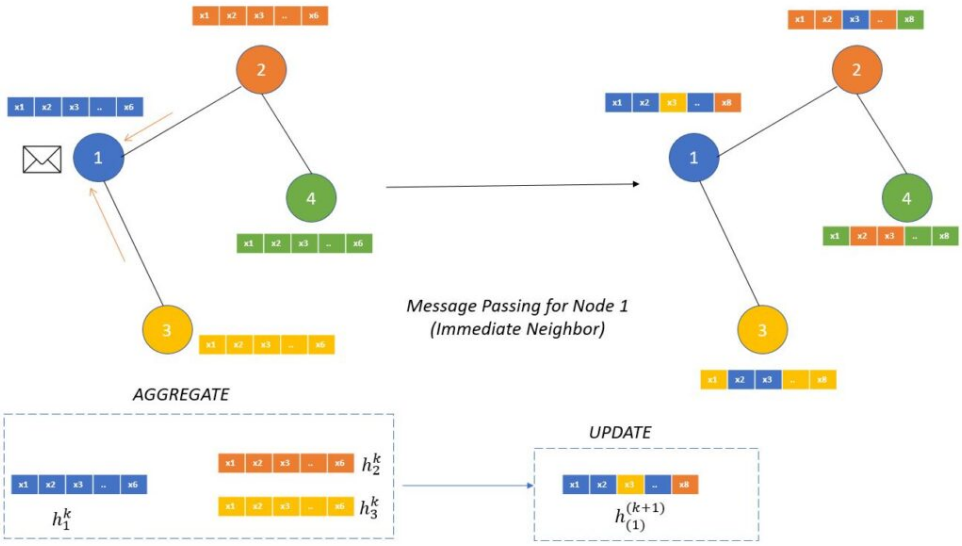


Figure 2.18. The message propagation process in GNNs [144].

GNNs are particularly effective at processing graph-structured data due to their ability to leverage both the individual features of nodes and the information encoded in the graph topology [139]. By iteratively aggregating information from neighboring nodes, GNNs can capture complex relationships and dependencies within the data [145]. This message-passing mechanism allows information to flow through the graph, enabling each node to collect and aggregate data from its local context—a crucial property for tasks such as recommendation, where relationships between users and items are paramount [143]. GNNs can perform various types of analyses on graphs, including node-level tasks (such as classification or prediction of individual properties), edge-level tasks (such as predicting the existence or attributes of links between nodes), and global graph-level tasks (such as classification of entire graphs). This flexibility makes them particularly well-suited for a wide range of recommendation scenarios, from predicting a user's preference for a specific item (link prediction in a user-item graph) to recommending groups of items or users (global tasks on the graph). Moreover, GNNs can reconstruct latent features solely from the graph structure, highlighting their ability to intrinsically extract information from relationships between entities [146].

2.5.3 Use of GNNs in Recommender Systems

In this section, we detail the use of Graph Neural Networks (GNNs) in the context of recommender systems, focusing primarily on the collaborative approach. The analysis presented is largely inspired by the article titled *Graph Neural Networks in Recommender Systems: A Survey* [121]. To better understand the operation and implementation of GNNs in recommender systems, we will introduce the associated mathematical formulations. For ease of reading and clarity of notation, a summary of the symbols used is presented in Table Table 2.1.

Notations	Descriptions
$\mathcal{U}/\mathcal{I}$	The set of users/items
A	Adjacency matrix of the graph
$\mathcal{N}_v$	Neighborhood set of node v
$\mathbf{h}_v^{(l)}$	Embedding of node v at layer l
$\mathbf{h}_i^{(l)}$	Embedding of item i at layer l
$\mathbf{n}_v^{(l)}$	Aggregated embedding of node v's neighbors at layer l
$\mathbf{h}_u^*$	Final embedding of user u
$\mathbf{h}_i^*$	Final embedding of item i
$\mathbf{b}^{(l)}$	Bias at layer l
$\oplus$	Vector concatenation

Table 2.1. Main notations used in this article [121].

In the context of GNN-based RS, users $u \in \mathcal{U}$ and items $i \in \mathcal{I}$ are typically modeled as nodes in a bipartite graph, where interactions between users and items are represented by edges, as illustrated in Figure 2.19.

The main task is to predict the preference level of a user u for an item i, denoted $y_{u,i}$. This variable can represent a binary value (interaction or not), an explicit rating (e.g., from 1 to 5), or a probability of interaction. In Figure 2.19, the preference level of User1 for Item2 is $y_{User1,Item2} = 3$.

To achieve this, each user and each item are associated with an embedding, denoted respectively as $\mathbf{h}_u^*$ and $\mathbf{h}_i^*$, obtained after neighborhood aggregation over multiple layers of the GNN.

The preference $y_{u,i}$ is then estimated using a function f that takes these two embeddings as inputs:

$$y_{u,i} = f\left(\mathbf{h}_u^*, \mathbf{h}_i^*\right) \tag{2.1}$$

The function f can take different forms depending on the model used. It can be

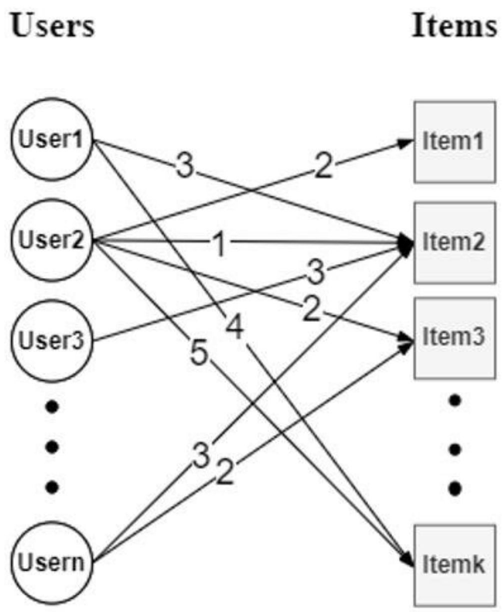


Figure 2.19. An example of a User-Item bipartite graph with ratings [147].

a dot product between the two vectors, a distance measure, or even a neural network (MLP) applied to the concatenation of both representations [148].

Mathematically, a graph $\mathcal{G}$ can be formalized by the pair $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ represents the set of nodes and $\mathcal{E}$ the set of edges, see Table Table 2.1. Let $v_i \in \mathcal{V}$ be a node and $e_{ij} = (v_i, v_j) \in \mathcal{E}$ an edge going from v_j to v_i. The neighborhood of a node v is then defined as $\mathcal{N}(v) = \{u \in \mathcal{V} \mid (v, u) \in \mathcal{E}\}$.

As previously mentioned, the fundamental principle of GNNs is based on an iterative process of aggregating the features of neighboring nodes, followed by integrating them into the current embedding of the central node during the propagation phase [149, 150]. From an architectural standpoint, a GNN is composed of several stacked propagation layers, each consisting of two main operations: aggregation of information from the neighborhood and updating the state of the central or target node. This propagation mechanism can be formalized by a specific equation, called the propagation formula, which implements these two steps recurrently across the model's layers. Here is the propagation formula:

$$\begin{aligned} \text{Aggregation:} \quad & \mathbf{n}_v^{(l)} = \text{Aggregator}_l \left(\left\{ \mathbf{h}_u^{(l)} \mid u \in \mathcal{N}_v \right\} \right), \\ \text{Update:} \quad & \mathbf{h}_v^{(l+1)} = \text{Update}_l \left(\mathbf{h}_v^{(l)}, \mathbf{n}_v^{(l)} \right), \end{aligned} \tag{2.2}$$

where $\mathbf{h}_u^{(l)}$ denotes the embedding of node u at the l-th layer, and Aggregator and Update represent the aggregation function and the update function at the l-th layer, respectively.

We briefly summarize below the aggregation and update operations of five typical GNN architectures that are widely used in the recommendation domain.

- **Graph Convolutional Networks (GCN)** [151] approximate the first-order spectral decomposition of the graph Laplacian to iteratively aggregate information from neighboring nodes.

- **GraphSAGE** [140] samples a fixed number of neighbors for each node and proposes various aggregation functions (mean, sum, max-pooling). The node representations are updated via concatenation:

$$\begin{aligned} \text{Aggregation:} \quad & \mathbf{n}_v^{(l)} = \text{Aggregator}_l\left(\{\mathbf{h}_u^{(l)}, \forall u \in \mathcal{N}_v\}\right) \\ \text{Update:} \quad & \mathbf{h}_v^{(l+1)} = \delta\left(\mathbf{W}^{(l)} \cdot \left[\mathbf{h}_v^{(l)} \oplus \mathbf{n}_v^{(l)}\right]\right) \end{aligned} \tag{2.3}$$

  where $\oplus$ denotes concatenation, $\delta(\cdot)$ is an activation function, and $\mathbf{W}^{(l)}$ is a learnable weight matrix.

- **Gated Graph Sequence Neural Networks (GGNN)** [152] incorporate a Gated Recurrent Unit (GRU) in the update step. This mechanism performs several recurrent passes over all nodes, which may lead to scalability issues when handling large graphs.

- **Hypergraph Neural Networks (HGNN)** [153] leverage a hypergraph structure to capture higher-order correlations among nodes.

- **Graph Attention Networks (GAT)** [154] assume that the influence of neighbors is neither uniform nor predetermined by the graph structure. An attention mechanism is used to assign a weight to each neighbor based on its features:

$$\begin{aligned} \text{Aggregation:} \quad & \mathbf{n}_v^{(l)} = \sum_{j \in \mathcal{N}_v} \alpha_{vj} \mathbf{h}_j^{(l)}, \quad \alpha_{vj} = \frac{\exp\left(\text{Att}(\mathbf{h}_v^{(l)}, \mathbf{h}_j^{(l)})\right)}{\sum_{k \in \mathcal{N}_v} \exp\left(\text{Att}(\mathbf{h}_v^{(l)}, \mathbf{h}_k^{(l)})\right)} \\ \text{Update:} \quad & \mathbf{h}_v^{(l+1)} = \delta\left(\mathbf{W}^{(l)} \mathbf{n}_v^{(l)}\right) \end{aligned} \tag{2.4}$$

  where $\text{Att}(\cdot)$ is an attention function (usually a fully connected layer followed by a non-linearity), α_{vj} is the normalized attention coefficient representing the importance of node j to node v, and $\mathbf{W}^{(l)}$ is a learnable weight matrix.

  To improve the model's stability and expressiveness, GAT employs *multi-head attentions*:

$$\mathbf{h}_v^{(l+1)} = \Big\Vert_{k=1}^{K} \delta\left(\sum_{j \in \mathcal{N}_v} \alpha_{vj}^k \mathbf{W}^k \mathbf{h}_j^{(l)}\right) \tag{2.5}$$

where K is the number of attention heads, and $\|$ denotes the concatenation operation.

Advantages of GAT:

- Ability to dynamically assign importance weights to neighbors, improving representation quality in heterogeneous or noisy graphs.
- Multi-head attention allows the model to capture various local dependencies and relations.
- Independence from the global graph structure; only local neighbor features are required for computing attention weights.

2.6 State of the Art of Recommender Systems in Healthcare

The scientific literature shows growing interest in the application of recommender systems across various healthcare domains [155]. These HRS (Health Recommender Systems) have been explored for their potential to personalize and improve multiple aspects of patient care, including medication recommendations, diagnostic assistance, treatment planning, and the provision of individualized lifestyle advice [156]. Although collaborative filtering has been applied in the medical field, it remains relatively less widespread as a standalone approach, in favor of content-based or knowledge-based methods [91]. Nevertheless, the shift toward hybrid approaches that combine collaborative filtering with other techniques represents a promising response to the inherent limitations of pure collaborative filtering, such as data sparsity or the cold-start problem [157]. At the same time, the integration of advanced machine learning methods, including deep learning–based collaborative filtering, is attracting increasing interest to enhance the personalization and accuracy of recommendations within HRS [156, 107].

After an in-depth analysis of several scientific articles addressing osteoarthritis and exercise-based management, it appears that few research efforts have been dedicated to developing recommender systems for physical exercise activities [158], and no study seems to specifically address the problem of personalized exercise recommendation for knee osteoarthritis patients using recommender system methods. Although the literature explores in detail the benefits of exercise in osteoarthritis management and examines various digital monitoring or intervention tools, no approach directly targets exercise personalization based on patient preferences or profiles through collaborative filtering techniques.

While collaborative filtering–based recommendation approaches have been tested to improve patient engagement and performance [159, 157, 156, 160], these works do not focus specifically on knee osteoarthritis. Furthermore, reviews dedicated to applying machine learning to the diagnosis and progression prediction of knee osteoarthritis are essentially limited to classification of radiographic images and prognostic modeling, without addressing therapeutic exercise recommendations [161, 162]. Similarly, recent studies evaluating the effectiveness of self-directed digital exercise interventions (mHealth apps) for osteoarthritis do not implement collaborative filtering or graph neural network mechanisms to personalize recommendations [163].

In the absence of a concrete example specifically targeting exercise recommendation for knee osteoarthritis via a collaborative filtering approach, our master's project represents the first research effort based on a collaborative filtering model enriched by Graph Neural Networks to propose personalized exercise programs.

2.7 Conclusion

This chapter has explored recommender systems, their main approaches (content-based, collaborative, and hybrid), and their growing importance in healthcare to personalize care. It has highlighted collaborative filtering, its variants, and the challenges related to data sparsity. The emergence of Graph Neural Networks (GNNs) was presented as a promising advancement to improve recommendation accuracy, although few works address their application to therapeutic exercise recommendation for gonarthrosis. This observation justifies the innovative approach developed in this project, detailed in the next chapter.

Chapter 3

Design of the Recommendation System

3.1 Introduction

This section presents the approach adopted to design a therapeutic exercise recommendation system for patients with gonarthrosis. This design is based on the specificities of the available biomechanical kinematic data as well as on the combined use of graph modeling and matrix factorization (MF) modeling.

The main objective is to develop an architecture capable of recommending physical exercises adapted to individual patient profiles, taking into account their clinical, functional, and biomechanical characteristics.

To model this problem, we have formulated the task as link prediction between two types of entities: patients and exercises. This leads us to represent the problem in the form of a bipartite graph, in which the nodes correspond to patients and exercises, and the edges indicate the actual prescription of an exercise to a patient. The challenge then consists of predicting new connections in this graph, that is, identifying which exercises could be recommended to a given patient.

The chosen solution relies mainly on GNNs, which allow the joint exploitation of relational structures between nodes and individual attributes associated with each entity. Additionally, we have modeled the solution in the form of MF in order to evaluate which of the two approaches offers the best results. Our objective is also to contribute to the advancement of scientific research in this field.

Thus, this design phase establishes the conceptual and technical framework on which the implementation of the models presented in the following chapter will be based.

3.2 Definition of Available Data

The database used in this work brings together a set of clinical, functional, and biomechanical information collected from patients with gonarthrosis. This data is essential for graph construction and personalization of recommendations. The database is an Excel file composed of five sheets, which we have separated into five files bearing the same names to facilitate their manipulation:

- **CinematicData.csv**: This contains 3D kinematic measurements of the knee recorded during the walking phase of patients. Each row represents a patient (413 individuals in total), and the columns correspond to knee joint variations along three axes: flexion-extension (*Flexion1* to *Flexion100*), abduction-adduction (*Abduction1* to *Abduction100*), and axial rotation (*Rotation1* to *Rotation100*). Each series of 100 columns represents the evolution of the joint angle throughout the walking cycle, divided into 100 regular intervals. For example, *Flexion1* corresponds to the flexion angle at the very beginning of the cycle (1%), *Flexion2* at 2%, and so on until *Flexion100* (100% of the walking cycle).

- **EmoviFeatures.csv**: This contains kinematic data that has been preprocessed and interpreted by the company *Emovi*. These variables correspond to clinical results derived from raw signals, reflecting the presence or absence of biomechanical anomalies.

- **70features.csv**: This file groups together a set of 70 numerical descriptors extracted from the two series of raw kinematic data. The data has been processed by an external organization, and we only use the provided results. This file allows reducing the raw kinematic data of dimension 300 (100 for flexion-extension, 100 for abduction-adduction, and 100 for rotation) and better characterizing the kinematic properties of joint movements.

- **Exercices + orthèses + MSK.csv**: This file contains the exercises recommended to patients (columns `ID_exo_1` to `ID_exo_14`), as well as certain relevant characteristics at the time of prescription, such as whether or not a knee brace is worn. For the complete list of these attributes, refer to appendix section 1.1. The other fields in this file group together the results of functional tests related to the biomechanics of the knee and lower limbs.

- **Liste détails 36 exercices.csv**: This file contains the exercises to be recommended. Each row corresponds to an exercise category, identified by a unique identifier. The `NewID` column represents the main identifier of each exercise category. For each category, the associated exercises are listed in columns `EXO_1` to `EXO_5`, which correspond to the specific identifiers of individual exercises belonging to this category. A brief description of each category is also included to better characterize the types of exercises grouped together.

One of the major challenges encountered in this work lies in the partial availability of data. Indeed, the database contains a significant number of missing values. This situation is largely explained by the inherent difficulty in collecting biological and functional data, which requires specialized equipment, rigorous protocols, and active patient involvement. Moreover, certain measurements are not systematically performed for all subjects, which limits the overall richness of the database and the representativeness of the profiles. This data scarcity imposes cautious methodological choices regarding the treatment of missing values, case exclusion, or variable selection for analysis.

3.3 Preprocessing and Structuring of Graph Data

The proper functioning of a GNN strongly depends on the quality and structuring of input data. In the context of this work, the data must be transformed into numerical representations that can be exploited by the model.

This section presents the necessary steps to encode node characteristics (patients and exercises), handle missing values, transform categorical variables, and structure relationships between entities in graph form. These operations constitute the preprocessing phase, essential for ensuring compatibility with the GNN architecture and guaranteeing the quality of learning.

For data preprocessing, here are the steps:

1. **Handling missing values**
   - For numerical columns, missing values are replaced by the column mean.
   - For categorical columns (text), they are replaced by the most frequent value (the mode).
2. **Handling outliers** [1] – Rather than removing aberrant values, these are capped: observations below the lower bound are brought back to this value,

[1] Outliers, or aberrant values, are observations that deviate significantly from the majority of data in a set. They can result from measurement errors, extreme natural variations, or rare events. Their detection is essential, as they can distort statistical analyses and predictive models if not properly handled.

and those above the upper bound are reduced to it. Here, the bounds are defined from the interquartile range (IQR):

$$\begin{aligned} \text{IQR} &= Q_3 - Q_1 \\ \text{lower bound} &= Q_1 - 1.5\text{IQR} \\ \text{upper bound} &= Q_3 + 1.5\text{IQR}. \end{aligned} \tag{3.1}$$

3. **Duplicate removal** – Duplicates are removed ensuring that each instance in the dataset appears only once.

4. **Standardization of numerical features** – Numerical features undergo a standardization process to put them on a common scale. This transformation adjusts the data so that they have a mean of zero and a standard deviation of one. For each value x in a feature, the standardized value z is calculated as follows:

$$z = \frac{x - \mu}{\sigma} \tag{3.2}$$

where μ is the mean of the feature and σ is its standard deviation. This step is crucial for algorithms sensitive to the scale of input features.

5. **Encoding categorical features** – Categorical variables (such as the affected knee or orthosis type) have been transformed by one-hot encoding [2].

algorithm 1 summarizes the data preprocessing pipeline.

[2]One-hot encoding is a technique for transforming categorical variables into binary vectors. Each category is represented by a vector where only one component equals 1 (indicating the presence of the category), while the others equal 0. For example, an *orthosis type* variable with three different modalities would be encoded into three binary columns.

Algorithm 1: Data Preprocessing

Input: Dataset D with numerical and categorical variables

Output: Preprocessed dataset D

foreach *numerical column col_num in D* **do**
    Replace missing values with the mean;
foreach *categorical column col_cat in D* **do**
    Replace missing values with the mode;
foreach *numerical column col_num in D* **do**
    Calculate Q_1, Q_3, IQR; Define *lower_bound*, *upper_bound*; **foreach** *value x in col_num* **do**
        if $x < lower\_bound$ **then**
            $x \leftarrow lower\_bound$;
        else
            if $x > upper\_bound$ **then**
                $x \leftarrow upper\_bound$;
Remove duplicates;
foreach *numerical column col_num in D* **do**
    Calculate μ, σ; **foreach** *value x in col_num* **do**
        $x \leftarrow \frac{x - \mu}{\sigma}$;
foreach *categorical column col_cat in D* **do**
    Apply one-hot encoding;
return *Preprocessed D;*

3.4 Design of the GNN Model

3.4.1 Description of the Entities

In the context of modeling the recommendation system with a GNN, two main types of nodes are defined: *patient* nodes and *exercise* nodes. The features of the patient nodes are extracted from all the files in the database, where each unique identifier (`ID_patient`) corresponds to a distinct node. As for the features of the exercise nodes, they are derived from the file `Liste détails 36 exercices`, which lists a set of 36 exercise categories, identified by the variable `New_ID`. Each category includes a set of exercises.

This schema allows for the construction of a bipartite graph, in which the edges connect patients to the exercises that have been prescribed to them. In other words, an edge is added between a patient and an exercise if the latter has been recommended to them; in the absence of a prescription, no edge is created. An illustration of this structure is presented in Figure 3.1.

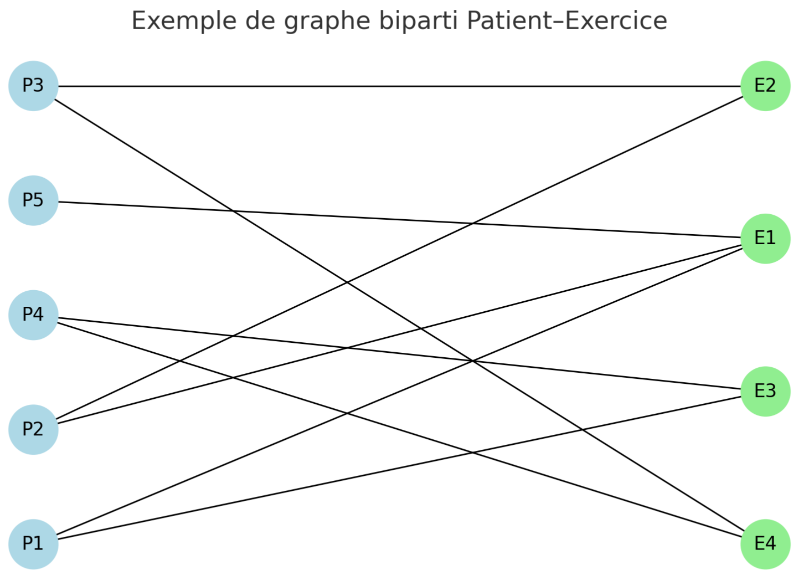


Figure 3.1. Visualization of a bipartite graph.

3.4.2 Feature Representation

In a GNN, each node is initially represented by a *feature* vector describing its properties. These vectors serve as the starting point for generating the embeddings learned through the layers of the GNN. Figure Figure 3.2 illustrates this fundamental concept.

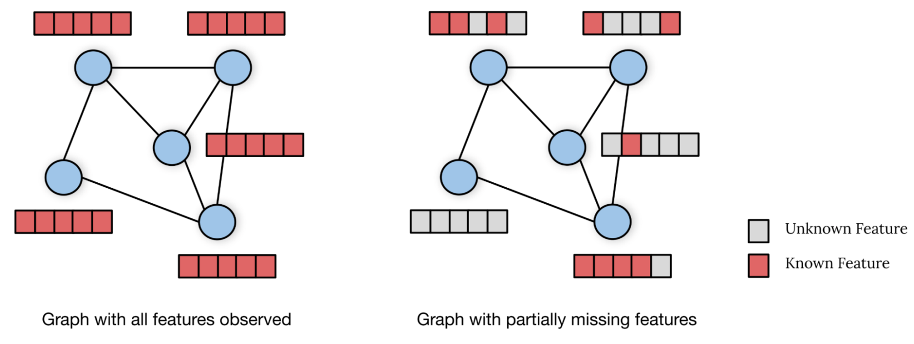


Figure 3.2. Illustration of the impact of initial features in a GNN [164].

When all features are available (left part of Figure 3.2), the GNN can directly exploit the properties of each node to propagate information through the graph. However, in many real-world cases, some features may be missing (right part of Figure 3.2), making learning more complex. In these cases, the GNN can fill in the gaps by leveraging the connections between nodes and learning from neighbors, highlighting the importance of the graph structure for enriching or reconstructing embeddings. Thus, the choice of initial features strongly influences the final performance of the model.

Features of "Patient" Nodes

The nodes representing patients in the graph are described using features extracted from the file *70Features.csv*. These features result from modifications made to the raw kinematic data. A file named *user_features.csv* was created and contains the attributes of the patient nodes. This file has dimensions 41371, corresponding to 413 patients, each with an identifier and 70 features.

Features of "Exercise" Nodes

The nodes representing exercises are described using information extracted from the "Description" column of the file *Liste détails 36 exercices.csv*. The approach is based on identifying keywords to generate binary features. For example, detecting the term "motor control" creates the variable *Motor control* (value 1 if present, 0 otherwise), and the presence of "hip flexor strengthening" creates the variable *Hip flexor strengthening*. Each keyword thus corresponds to a binary feature.

A file named `item_features.csv` was created, with 71 rows (exercises) and 38 columns: the exercise identifier and 37 extracted features.

3.4.3 Definition of Graph Edges

The graph edges represent the prescription relationship between a patient and an exercise. More precisely, a link is created when an exercise is prescribed to a patient. These relationships have been extracted from columns *ID_exo_1* to *ID_exo_14* of the file *Exercices + orthèses + MSK.csv*, as illustrated in Figure 3.1.

The graph integrates two types of bidirectional relationships:

1. **"Recommends" relationship**: patient $\xrightarrow{\text{recommends}}$ exercise.
2. **"Recommended to" relationship**: exercise $\xrightarrow{\text{recommended to}}$ patient.

This bidirectional structure allows information propagation in both directions, enriching the representation of embeddings through neighborhood aggregation.

3.4.4 GNN Model Architecture

The proposed GNN architecture follows a hierarchical multi-layer approach, combining the advantages of graph convolutions and attention mechanisms to capture complex patient-exercise interaction patterns.

Before propagation, the input features (user and item features) are projected into a latent space of 256 dimensions using linear layers. After three successive GAT layers, the final embeddings are reduced to 128 dimensions.

Residual connections stabilize training and, after each layer (except the last), the following are applied successively:

1. Batch Normalization[3],
2. the ReLU activation function to introduce non-linearity,
3. dropout[4] configured dynamically on features and edges. This technique regularizes the model and helps reduce overfitting[5].

3.4.5 Loss Function

The loss function used in this work is the *Focal Loss*, defined by the following equation:

$$\mathcal{L}_{\text{focal}} = -\alpha(1-p_t)^{\gamma}\log(p_t) \tag{3.3}$$

where α and γ are weighting hyperparameters that modulate the impact of examples on the total loss. The term p_t represents the predicted probability for the correct class and is defined by:

$$p_t = \begin{cases} p & \text{if the true class is 1,} \\ 1-p & \text{if the true class is 0,} \end{cases} \tag{3.4}$$

[3] *Batch Normalization* [165] aims to accelerate convergence and stabilize training by normalizing, for each mini-batch, the activations of each neuron:

$$\hat{x}_i = \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \varepsilon}}, \quad y_i = \gamma\,\hat{x}_i + \beta,$$

where γ and β are learned parameters, and ε is a numerical regularization term.

[4] *Dropout* consists of randomly deactivating, at each training iteration, a fraction p of the neurons in a layer. For each neuron, a Bernoulli variable $d_i \sim \text{Bernoulli}(1-p)$ is sampled and its activation x_i is replaced by $d_i x_i$. During inference, no suppression is performed, but activations are multiplied by $1-p$ to compensate for the dropout effect.

[5] Overfitting is a phenomenon where a model adapts too strongly to training data, capturing even noise or specific particularities, which hurts its ability to generalize to new data.

with p being the probability estimated by the model for the positive class. Thus, p_t always represents the probability assigned by the model to the correct target class, regardless of its actual value (0 or 1).

The *Focal Loss* was chosen to mitigate the impact of class imbalance, which is frequent in recommendation systems where certain user-item interactions are rare. The term $(1-p_t)^\gamma$ reduces the influence of easy-to-classify examples, focusing learning on difficult cases. This mechanism strengthens the model's ability to handle minority classes, thus improving the quality and relevance of the produced recommendations.

3.5 Design of the Matrix Factorization Model

The dataset used for this therapeutic exercise recommendation system includes the same 413 unique patients and 71 different exercises as those used in the GNN model, generating a total of 9,376 user-item interactions. The exercises recommended to each patient were extracted from the file `Liste détails 36 exercices.csv`, allowing the construction of a binary user-exercise matrix: a value of 1 indicates that an exercise was recommended to a given patient, while a value of 0 means no recommendation.

An illustrative excerpt of this matrix is presented in Table 3.1 below:

user-item	Item 1	Item 2	Item 3	Item 4
User 1	1	0	1	0
User 2	0	1	1	0
User 3	1	0	0	1

Table 3.1. Example of binary user-item sub-matrix extracted from the data

In this example, exercise 1 was recommended to patients 1 and 3, while exercise 2 was only prescribed to patient 2. This binary representation constitutes the main input for training the matrix factorization model. For evaluation, the data was divided according to a stratified 80/20 partition, allowing for a reliable measure of the model's generalization performance on unseen data.

3.5.1 Principle of Matrix Factorization

Matrix Factorization consists of decomposing the binary user-item interaction matrix $R \in \{0,1\}^{nm}$ into two real matrices of lower rank $W \in \mathbb{R}^{nk}$ and $H \in \mathbb{R}^{km}$ where $k \ll \min(n,m)$ is the number of latent factors.

Each user u is thus represented by a vector $w_u \in \mathbb{R}^k$ (row of W), and each exercise i by a vector $h_i \in \mathbb{R}^k$ (column of H).

The prediction of the probability that a user u will be recommended exercise i is performed via a biased dot product:

$$y_{u,i} = \sigma\big(w_u^\top h_i + b_u + b_i + b_0\big),$$

where b_u (resp. b_i) is the bias associated with user u (resp. with exercise i), b_0 a global bias, and $\sigma(x) = \frac{1}{1+e^{-x}}$ the sigmoid function. The recommendation is made according to a defined threshold.

3.5.2 Loss Function

To learn W, H and the biases, we minimize the *binary cross-entropy* between the predictions $\hat{r}_{u,i}$ and the observed values $r_{u,i} \in \{0, 1\}$. Denoting Ω as the set of pairs (u, i) used (balanced sampling of positives and negatives), the loss is written as:

$$\mathcal{L} = -\frac{1}{|\Omega|} \sum_{(u,i)\in\Omega} \Big[r_{u,i} \log \hat{r}_{u,i} \; + \; (1 - r_{u,i}) \log(1 - \hat{r}_{u,i})\Big]. \tag{3.5}$$

This formulation symmetrically penalizes errors on positive and negative examples, and naturally adapts to the binary classification task of recommendation / non-recommendation.

In summary, MF learns user and exercise *embeddings* of dimension k which, combined with bias, provide an estimation of the probability of each interaction, optimized by minimizing the binary cross-entropy.

3.6 Cold Start Problem

The *cold start problem* refers to the difficulty encountered by recommendation systems when they must handle new users or new items for which no prior interaction data is available. In the context of collaborative approaches, particularly graph-based models, this absence of history prevents the direct computation of relevant embeddings, which limits the model's ability to generate adapted recommendations.

To address this limitation, a deep neural network model, which we have named the *Embedding Encoder*, has been designed to project user characteristics into the embedding space learned by the GNN or MF model. This model is trained to reproduce the embeddings generated by the GNN or MF from data of patients already present in the graph, thus ensuring the consistency of embeddings. When a new user is encountered, their characteristics (70-dimensional vector) are transformed into an embedding compatible with the latent space of the GNN or MF. The recommendation system can then calculate similarity scores between this projected embedding and the item embeddings already learned by the GNN or MF, allowing the generation of a list of the most suitable exercises for this new user.

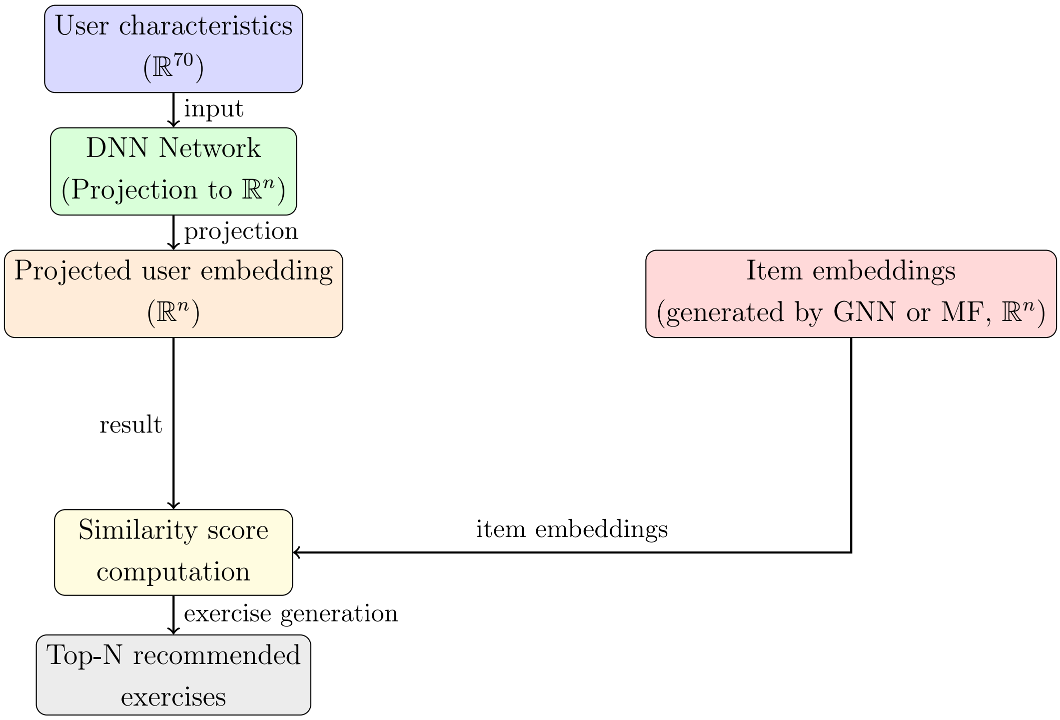


Figure 3.3. Diagram of Cold Start problem handling.

The diagram illustrates the strategy adopted to solve the cold start problem. The characteristics of new users are projected into the embedding space learned by the GNN or MF using the Embedding Encoder. The obtained embedding is then compared to the embeddings of existing items to generate adapted recommendations.

3.7 Comparison of GNN and MF Approaches for Recommendation

In order to better understand the respective specificities, advantages and limitations of the two approaches studied in this work, namely Graph Neural Networks (GNN) and Matrix Factorization (MF), Table 3.2 presents a comparative synthesis according to several key criteria influencing the quality, interpretability and complexity of the generated recommendations.

Criterion	**GNN**	**MF**
Exploits features	Yes	No
Requires a graph	Yes	No
Sensitive to *cold start*	Yes	Yes
Computational complexity	Higher	Lower
Quality of learned embeddings	Higher	Medium

Table 3.2. Comparison of GNN and Matrix Factorization approaches.

3.8 Conclusion

This chapter has presented all the design steps of the exercise recommendation system for patients with gonarthrosis, based on two approaches: graph modeling via Graph Neural Networks (GNN) and Matrix Factorization (MF).

The GNN is distinguished by its ability to exploit both the individual attributes of nodes and the relationships between them, thus providing rich and contextualized latent representations. Conversely, MF allows efficient modeling of direct interactions in the form of latent factors, with lower complexity but at the cost of reduced exploitation of the intrinsic characteristics of entities.

In order to overcome the critical cold start problem, an Embedding Encoder model has been integrated to project the characteristics of new patients into the latent space learned by each model. This approach guarantees the generalizability of the system, even for profiles never observed during training.

The following chapter will detail the practical implementation of these models, the technical choices made as well as the comparative evaluation of the performances obtained on the clinical dataset.

Chapter 4

Implementation and Testing

4.1 Introduction

This chapter presents the practical implementation and experimental evaluation of our rehabilitation exercise recommendation system based on GNNs and MF. The main objective is to develop, optimize and validate a model capable of generating personalized recommendations for therapeutic exercises by exploiting the complex relationships between patients' pathological profiles and the biomechanical properties of exercises.

The implementation phase is structured around several fundamental axes: the development environment, decoding and prediction mechanisms, adopted training strategies, including hyperparameter optimization and regularization techniques implemented to maximize performance while preserving generalization capability. The experimental evaluation is based on a methodology of systematic exploration of the hyperparameter space, allowing the identification of optimal configurations for GNN and MF architectures. This comparative analysis relies on the *Recommender Performance Score* (RPS), a composite metric specifically designed to evaluate the overall quality of recommendation systems in medical contexts.

4.2 Development Environment

This project required a development environment adapted to the complexity of heterogeneous graphs and intensive training of Graph Neural Network (GNN) models. Two distinct workstations were used for development, training and model evaluation.

4.2.1 Hardware Used

We primarily used two workstations to perform calculations and model training. Table 4.1 presents the technical characteristics of these workstations used.

Component	Workstation 1	Workstation 2
Processor	Intel Core i5-12400f	Apple M1 (ARM-based)
RAM	16 GB	16 GB
GPU	NVIDIA RTX 3050 (CUDA 11.8)	Apple M1 GPU
Operating System	Windows 11	macOS Sequoia 15.4.1
Tools	VS Code, Jupyter Notebook	VS Code, Jupyter Notebook

Table 4.1. Characteristics of the workstations used

4.2.2 Software and Libraries

Development was carried out in **Python 3.13.3**, with the support of the following libraries:

- **PyTorch**: for creating GNN layers and model training;
- **PyTorch Geometric**: efficient manipulation of heterogeneous graphs;
- **Scikit-learn**: data preprocessing, metric computation;
- **Pickle**: serialization and saving of Python objects (scaler, embeddings, etc.);
- **Pandas** and **NumPy**: management of tabular datasets;

4.3 Decoding and Score Prediction for GNN

The recommendation score $y_{u,i}$ for a pair (u, i) is calculated by combining three distinct approaches:

- **MLP**: a three-layer network that transforms the concatenation of user and exercise embeddings $\left[e_u \,\|\, e_i\right] \in \mathbb{R}^{2d}$ (with $d = 128$) into a single score S_{MLP}. This concatenation allows joint exploitation of the characteristics of both nodes via a non-linear transformation learned by the network.
- **Dot product**: $S_{\text{dot}} = e_u^\top e_i$, which measures the direct similarity of vectors in the latent space. A high score indicates strong affinity.

- **Cosine similarity**: $S_{\cos} = \frac{e_u}{\|e_u\|_2} \cdot \frac{e_i}{\|e_i\|_2}$, which evaluates the directional alignment of vectors, regardless of their norm.

The MLP model follows the following architecture:

0. **First linear layer** $(2d \to d)$ with ReLU activation and Dropout:

$$h_1 = \mathrm{ReLU}(W_1 x + b_1), \quad h'_1 = \mathrm{Dropout}(h_1) \tag{4.1}$$

0. **Second linear layer** $(d \to d/2)$ with ReLU:

$$h_2 = \mathrm{ReLU}(W_2 h'_1 + b_2) \tag{4.2}$$

0. **Third linear layer** $(d/2 \to 1)$ producing the final score:

$$S_{\mathrm{MLP}} = W_3 h_2 + b_3 \tag{4.3}$$

The total recommendation score is a weighted average of the following three components:

$$y_{u,i} = 0.5\, S_{\mathrm{MLP}} + 0.3\, S_{\mathrm{dot}} + 0.2\, S_{\cos} \tag{4.4}$$

These weights were determined empirically to maximize performance on the validation set.

The choice to assign a main weight of 0.5 to the score produced by the MLP is based on its ability to capture complex non-linear relationships between patient and exercise characteristics, thus offering better expressivity and greater adaptability to data specificities. This component is considered the most determining factor in the final prediction.

The dot product receives an intermediate weight of 0.3 because it constitutes a simple, fast and effective method for evaluating overall similarity between latent representations of nodes. It acts as a robust estimator of compatibility between entities, particularly when embeddings already contain relevant information extracted by the GNN model.

Finally, cosine similarity is weighted at 0.2, as it provides a normalized measure of directional alignment between embedding vectors, which allows complementing the other two scores by capturing additional geometric information, particularly when vector amplitudes may vary without affecting the direction of similarity.

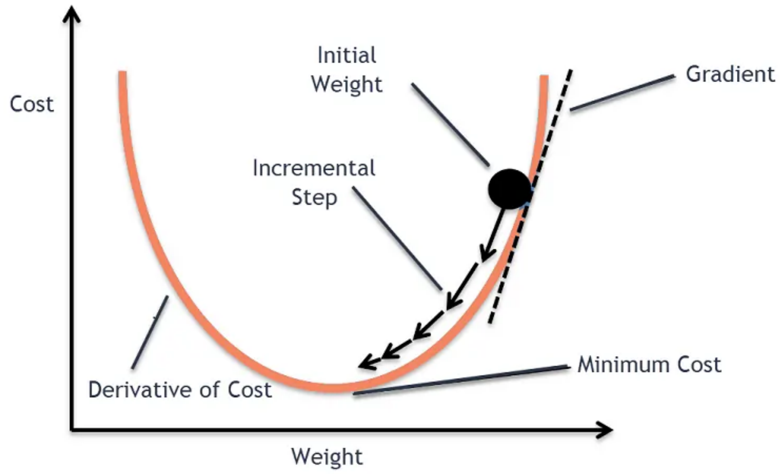


Figure 4.1. The gradient descent algorithm [166].

4.4 Model Training

4.4.1 GNN Model Training

GNN model training is based on an optimization algorithm based on the gradient descent algorithm, illustrated in Figure 4.1.

Training uses the *Adam* optimizer, accompanied by an adaptive scheduler to reduce the learning rate in case of validation metric stagnation. Graph edges are randomly separated into training and test sets via a custom function.

To counter class imbalance, *negative sampling*[1] is employed with a fixed ratio of 2, producing two negatives for each positive.

The loss function used is the *Focal Loss*. This formulation allows modulating the importance of easy and difficult examples, by reducing the influence of well-classified examples (high p_t) and emphasizing those that are poorly predicted (low p_t). *Gradient clipping* [2] is applied to prevent gradient explosion.

4.4.2 Embedding Encoder Model Training

As a reminder, the *Embedding Encoder* model aims to project clinical data from new patients into the embedding space learned by the GNN or MF model. Since

[1] For each positive interaction, we generate r negative examples by randomly sampling user-item pairs absent from real interactions.

[2] *Gradient clipping* is a technique used during neural network training to limit the norm of gradients calculated during backpropagation. When gradients exceed a specified threshold value, they are reduced (*clipped*) to avoid gradient explosion problems, thus stabilizing learning, particularly in deep or recurrent models.

the latent spaces generated respectively by the GNN and MF differ, two distinct *Embedding Encoder* models are trained separately.

GNN Space

For the *Embedding Encoder* associated with the GNN space, the evolution of the loss function during training is illustrated in Figure 4.2.

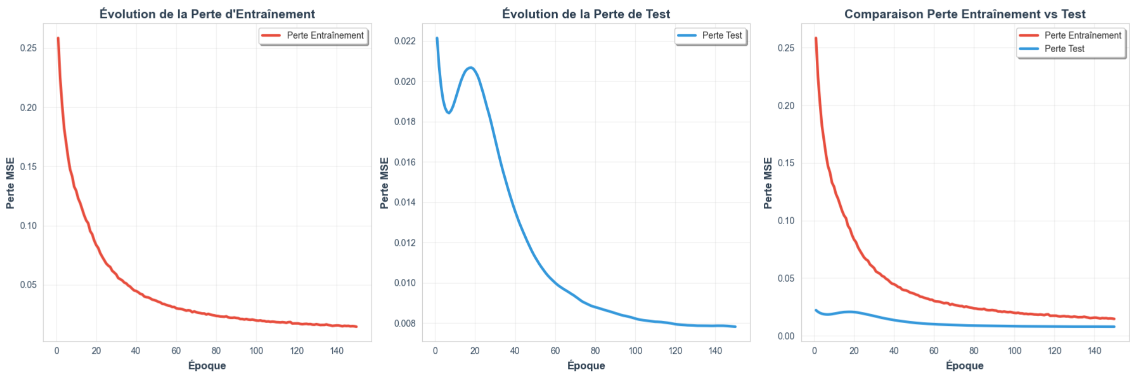


Figure 4.2. Evolution of the loss function during Embedding Encoder training in GNN space.

The final root mean square error (RMSE) obtained on the validation set is 0.0314, demonstrating excellent model capacity to faithfully approximate patient embeddings. This performance guarantees relevant and reliable recommendations, even in cold start situations, i.e., for patients who have never interacted with the system.

MF Space

Similarly, a second *Embedding Encoder* model is trained to project new patients into the latent space derived from MF. The evolution of the loss function associated with this training is presented in Figure 4.3.

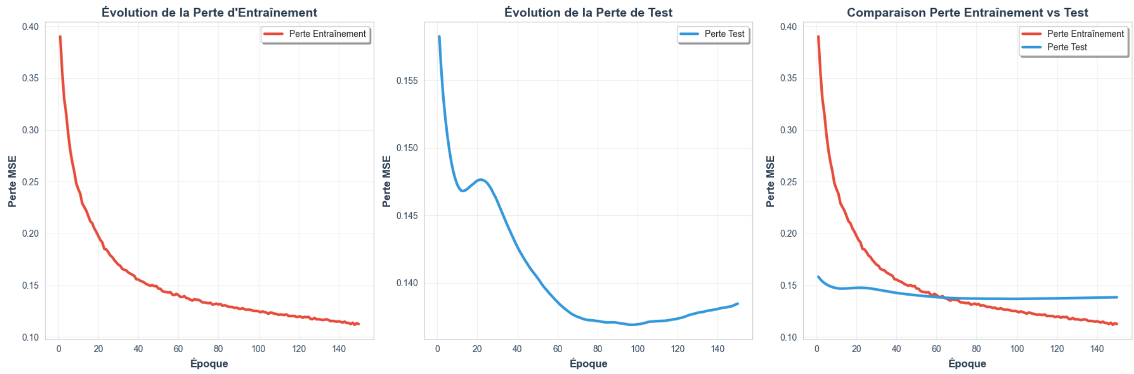


Figure 4.3. Evolution of the loss function during Embedding Encoder training in MF space.

In this case, the RMSE obtained on the validation set is 0.1755, a value slightly higher than that observed for the GNN space. This difference is explained by the larger dimension of the latent space generated by MF, making embedding approximation more complex.

Nevertheless, the quality of this approximation remains sufficient to ensure consistency of recommendations produced for new patients, in line with the latent representations learned during MF model training.

Thus, the joint use of two distinct *Embedding Encoder* models allows specific adaptation to the structure of each latent space (GNN or MF), guaranteeing robust and clinically relevant recommendations even in *cold start* situations.

4.5 Performance Evaluation of GNN and MF Models

Model performance is measured by the *Recommender Performance Score (RPS)*, a composite metric defined as a weighted combination of three essential evaluation measures:

- **AUC (Area Under the Curve)** (weight 0.5): measures the model's ability to distinguish between positive classes (existing links) and negative classes (non-existing links), as illustrated in Figure 4.4. A value of 1.0 indicates perfect separation, while a value of 0.5 corresponds to a random model.

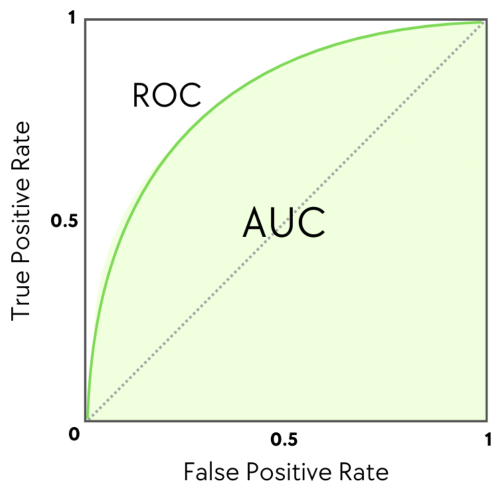


Figure 4.4. The AUC plot [167].

- **AP (Average Precision)** (weight 0.3): reflects the average precision of model predictions across all possible thresholds, which allows evaluating the model's ability to correctly classify positive examples at the top of the recommendation list.

- **Accuracy** (weight 0.2): indicates the overall rate of correct classifications, i.e., the proportion of correctly predicted examples, all classes combined.

The values in parentheses correspond to the weights assigned to each metric in the RPS calculation. The complete formula is written as:

$$\text{RPS} = 0.5\text{AUC} + 0.3\text{AP} + 0.2\text{Accuracy} \tag{4.5}$$

The RPS was designed to provide a synthetic and balanced measure of recommendation model performance by combining several complementary aspects:

- **AUC** receives the highest weight (0.5) because it evaluates the overall discriminative capacity of the model, independent of threshold choice.

- **Average Precision (AP)** is weighted at 0.3 because it focuses on the quality of the first recommendations. Moreover, AP is particularly adapted to contexts where classes are imbalanced, as in recommendation tasks where positive interactions represent a minority compared to the set of all possible pairs. This metric thus allows evaluating the model's ability to effectively identify relevant elements without being influenced by the negative majority, unlike measures such as Accuracy.

- **Accuracy** is included with a reduced weight (0.2) because, although representative of the overall rate of correct predictions, it can be misleading in the presence of imbalanced classes, which is frequent in link prediction or recommendation tasks.

The combined use of these three metrics avoids biases related to a single measure and provides a more complete and representative evaluation of the model's real performance, both on its overall ranking capability and on the quality of recommendations and prediction reliability.

Table 4.2 allows interpreting the RPS value and evaluating the overall quality of model performance.

RPS Score	Quality	Interpretation	Justification
≥ 0.80	Excellent	Highly reliable system, capable of producing relevant and coherent recommendations.	Good balance between precision, ranking capability and robustness to data variations.
≥ 0.75	Very good	Performing model with recommendation error risk.	Good compromise between generalization and specialization.
≥ 0.70	Good	Functional system; however, adjustments could strengthen its robustness.	Performance is correct but the model may fail on certain data subgroups.
≥ 0.60	Acceptable	Usable for experimental or research purposes.	The model presents notable gaps in classification or recommendation precision.
< 0.60	Insufficient	Model unsuitable for recommendation objectives.	Poor generalization and ranking capability; performance comparable to a random or naive model.

Table 4.2. RPS score interpretation grid

4.6 Search for Optimal GNN Model Configuration

4.6.1 Experimental Objectives

The objective of this phase is to determine the optimal combination of GNN model hyperparameters to maximize RPS. Several test series were conducted by varying an exhaustive list of GNN hyperparameters, such as hidden dimensions, embedding dimensions, as well as the number of heads in attention layers. Hyperparameters are divided into two categories: model hyperparameters and those related to training.

Model Hyperparameters

- **Hidden dimension** (`hidden_dim`) – Corresponds to the number of neurons in each intermediate layer of the GNN. These neurons are responsible for processing and transforming user and item characteristics.

- **Embedding dimension** (`embedding_dim`) – Represents the size of the latent vector space into which users and items are projected. This is the dimension of the embedding vector used to encode node characteristics in a common space.

- **Number of layers** (`num_layers`) – Indicates the total number of GNN convolution layers.

- **Dropout rate** (`dropout`) – Rate of random neuron deactivation during training, used to regularize the model and limit overfitting.

- **Residual connections** (`use_residual`) – Residual connections allow adding a layer's input to its output, facilitating learning in deep networks by preserving initial information. The output of a layer with residual connection is given by:

  $$\mathbf{y} = \mathcal{F}(\mathbf{x}) + \mathbf{x} \tag{4.6}$$

  where $\mathbf{x}$ is the layer input, $\mathcal{F}(\mathbf{x})$ the transformation performed (e.g., convolution followed by activation) and $\mathbf{y}$ the final output. This mechanism improves training stability and prevents gradient attenuation in deep architectures, as in our case.

- **Batch normalization** (`use_batch_norm`) – *Batch Normalization* is a training technique introduced by [165] aimed at stabilizing and accelerating the learning of deep neural networks. It consists of normalizing layer activations by applying a statistical transformation that forces their mean to zero and their variance to one. Formally, for a mini-batch $\mathcal{B} = \{x_1, x_2, ..., x_m\}$, normalization is performed as follows:

$$\hat{x}_i = \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \varepsilon}}, \tag{4.7}$$

where $\mu_{\mathcal{B}}$ and $\sigma_{\mathcal{B}}^2$ are respectively the mean and variance of the mini-batch, and ε a regularization term to avoid division by zero. Two learned parameters, γ (scale factor) and β (offset), allow the layer to recover the identity if necessary:

$$y_i = \gamma\hat{x}_i + \beta. \tag{4.8}$$

- **Attention mechanism** (`use_attention` and `attention_heads`) – The attention mechanism in GNNs allows the model to give more importance to certain relationships in the graph. The `attention_heads` parameter determines the number of heads used in multi-head attention, each head learning to capture a specific aspect of relationships between nodes.
- **MLP decoder** (`simple_decoder`) – Indicates whether the GNN decoder uses a simplified architecture (dot product) or a more expressive architecture based on a multilayer perceptron (MLP).

Training Hyperparameters

The results of training hyperparameter tests are found in **??**.

- **Initial learning rate** (`lr = 0.005`) – It controls the size of steps taken when updating model weights at each training iteration.
- **Learning rate scheduler** – A `ReduceLROnPlateau` is used to dynamically adjust the learning rate based on validation set performance, it accepts 3 parameters:
  - **Mode** `mode = 'max'` – The scheduler expects metric maximization (here RPS).
  - **Reduction factor** (`factor` = 0.75) – In case of RPS stagnation, the learning rate is multiplied by this factor.
  - **Patience** (`patience` = 10) – If no RPS improvement is observed for 10 consecutive epochs, the learning rate is reduced.
  - **Minimum rate** (`min_lr` = 10^{-6}) – The scheduler will never lower the learning rate below this value.
- **Negative sampling ratio** (`neg_ratio` = 2) – Controls the negative sampling ratio during recommendation model training.

4.6.2 Experimental Results and Analysis

Below are the results obtained after model training according to different hyperparameter configurations.

Hidden Dimension and Embedding Dimension

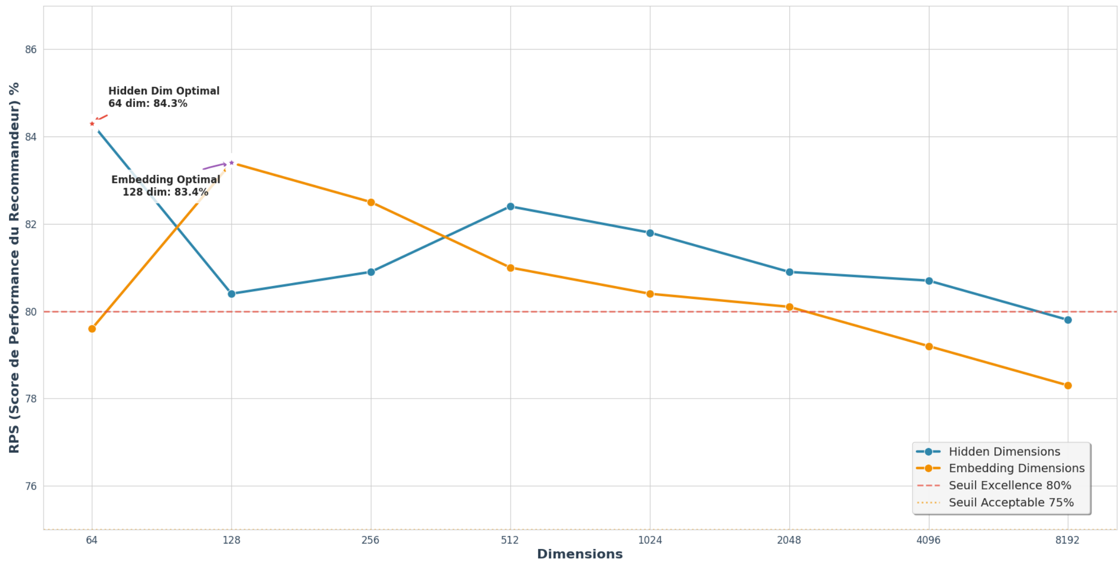


Figure 4.5. Evolution of RPS as a function of `hidden_dim` and `embedding_dim`.

Experiments on hyperparameters `hidden_dim` and `embedding_dim` (64 to 8192) show that the model achieves its best performance with `hidden_dim` = 64 (84.3% RPS) and `embedding_dim` = 128 (83.4% RPS). This optimal configuration (64 hidden dimensions, 128 embedding dimensions) was retained for the final model, offering a good balance between performance and computational efficiency.

Number of Layers

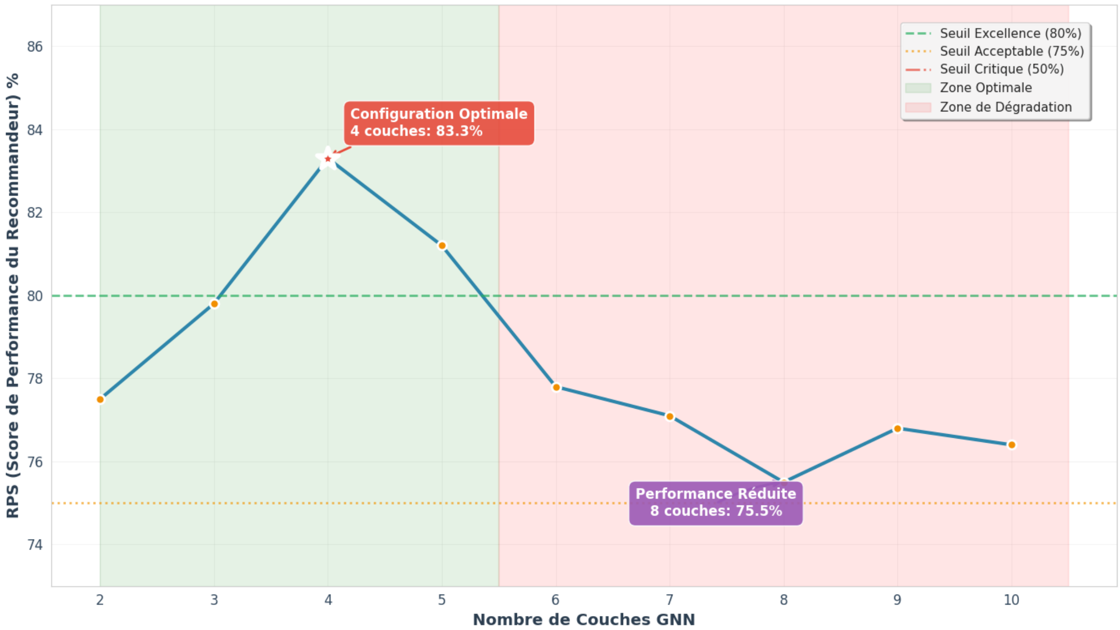


Figure 4.6. Evolution of RPS as a function of the number of layers (`nb_layers`).

Analysis of model depth reveals an optimal four-layer architecture achieving 83.3% RPS. With two layers, performance remains limited (77.5%) by insufficient extraction of structural patterns[3] present in the graph. Beyond five layers, moderate degradation appears (75.5% for eight layers) due to progressive homogenization of nodal representations that decreases the model's discriminative power.

[3] *Structural patterns* represent characteristic recurrent substructures in the graph, such as hubs, bridges or paths, which translate important relationships between nodes. In the recommendation context, they allow capturing usage similarities or shared behaviors between user or item groups.

Dropout

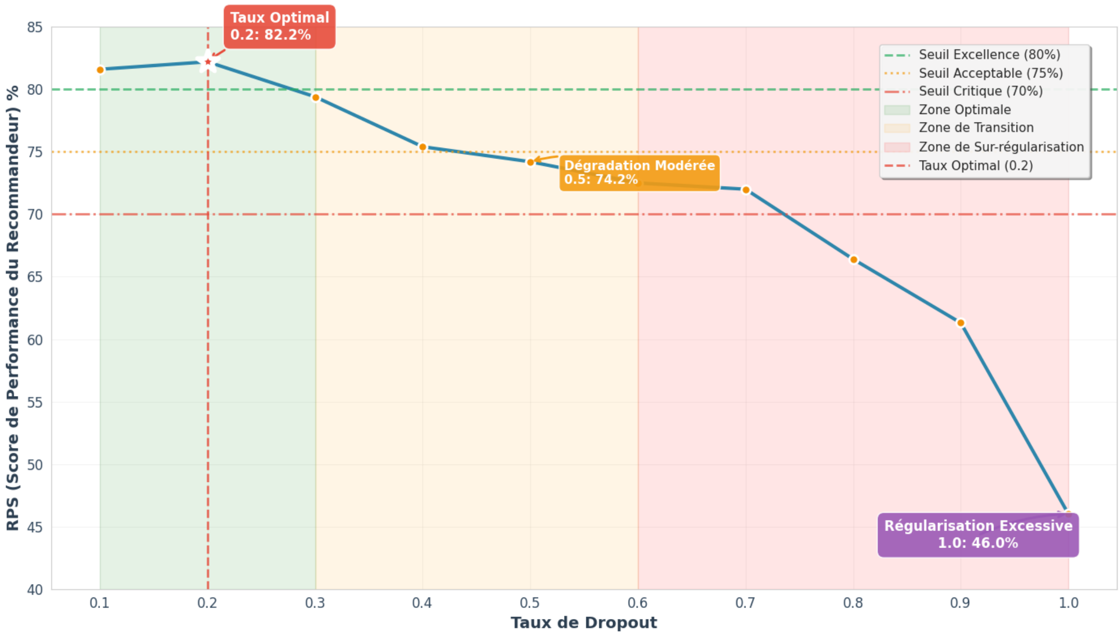


Figure 4.7. Evolution of RPS as a function of dropout.

The dropout study reveals an optimal configuration at 0.2 achieving 82.2% RPS. Analysis shows three distinct zones: an optimal zone (0.1 to 0.3) with performance above 79%, a transition zone (0.3 to 0.6) with progressive degradation, and an over-regularization zone (beyond 0.6) culminating at 46% RPS for dropout of 1.0, a degradation of 36.2% compared to the optimal configuration.

Attention Mechanism

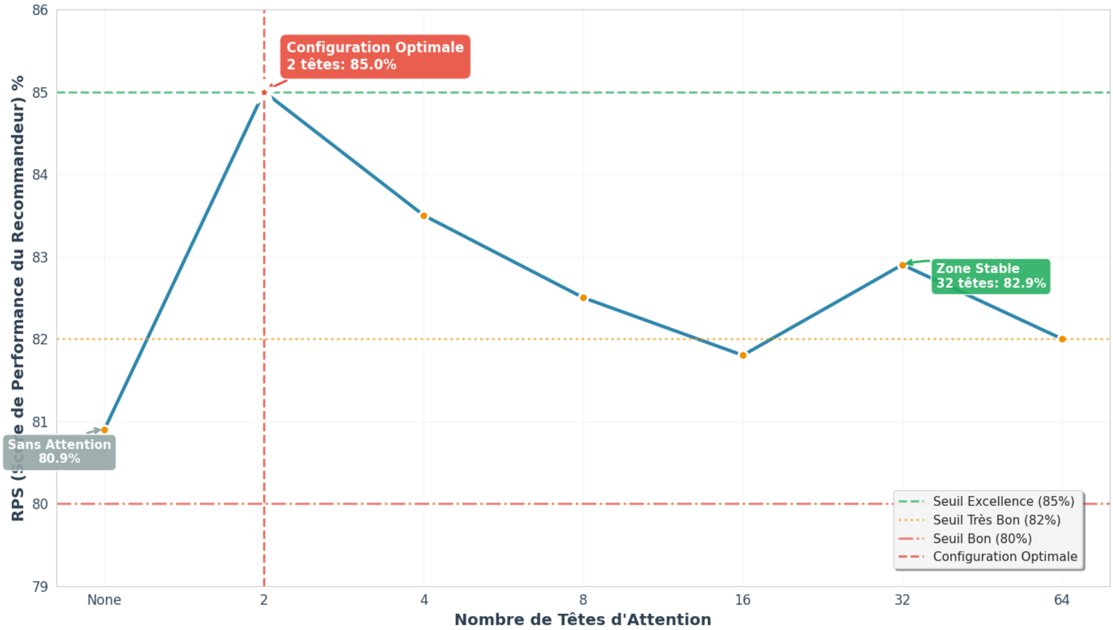


Figure 4.8. Evolution of RPS as a function of attention mechanism use and number of attention heads.

Integration of the attention mechanism via *Graph Attention Networks (GAT)* brings a substantial gain of +4.1% RPS, going from 80.9% (base configuration) to 85.0% with multi-head attention. The optimal configuration is achieved with only 2 attention heads, performance decreasing gradually beyond (83.5% for 4 heads, then stabilization between 81.8% and 82.9% for 8 to 64 heads), demonstrating that a minimalist approach is more effective in our context.

Architectural Components

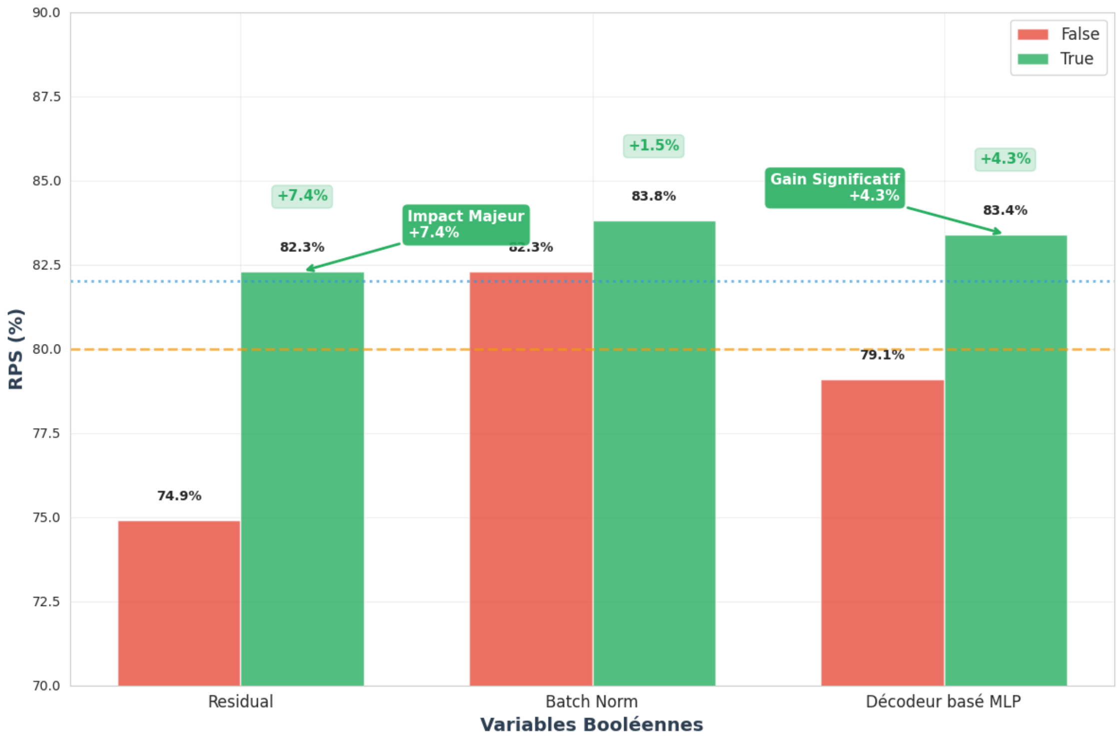


Figure 4.9. Evolution of RPS according to the use boolean parameters.

Analysis of boolean components reveals three cumulative positive contributions: residual connections bring the most significant gain (+7.4%, from 74.9% to 82.3%) by attenuating vanishing gradient, batch normalization provides a moderate gain (+1.5%, from 82.3% to 83.8%) by stabilizing training, and the MLP decoder generates a significant gain (+4.3%, from 79.1% to 83.4%) by capturing complex non-linear interactions. The optimal configuration combines all three activated components.

4.6.3 Optimal Configuration

The optimal configuration of the GNN model, determined after thorough hyperparameter exploration, is summarized in table Table 4.3, and the generated GNN is associated with a total of 132,545 trainable parameters.

Hyperparameter	**Optimal value**
Hidden dimension (`hidden_dim`)	64
Embedding dimension (`embedding_dim`)	128
Number of GNN layers (`num_layers`)	4
Dropout rate (`dropout`)	0.2
Residual connections (`use_residual`)	Yes
Batch Normalization (`use_batch_norm`)	Yes
Attention mechanism (`use_attention`)	Yes (2 heads)
Decoder type (`simple_decoder`)	MLP-based decoder
Learning rate (`learning_rate`)	0.005
Scheduler patience (`patience`)	15
LR reduction factor (`factor`)	0.75
Minimum learning rate (`min_lr`)	110^{-6}
Focal Loss parameter γ	3.0
Focal Loss parameter α	0.25
Negative sampling ratio (`neg_ratio`)	2

Table 4.3. Hyperparameters of the best model configuration

The evolution of learning metrics for the model according to this optimal configuration is illustrated in Figure 4.10.

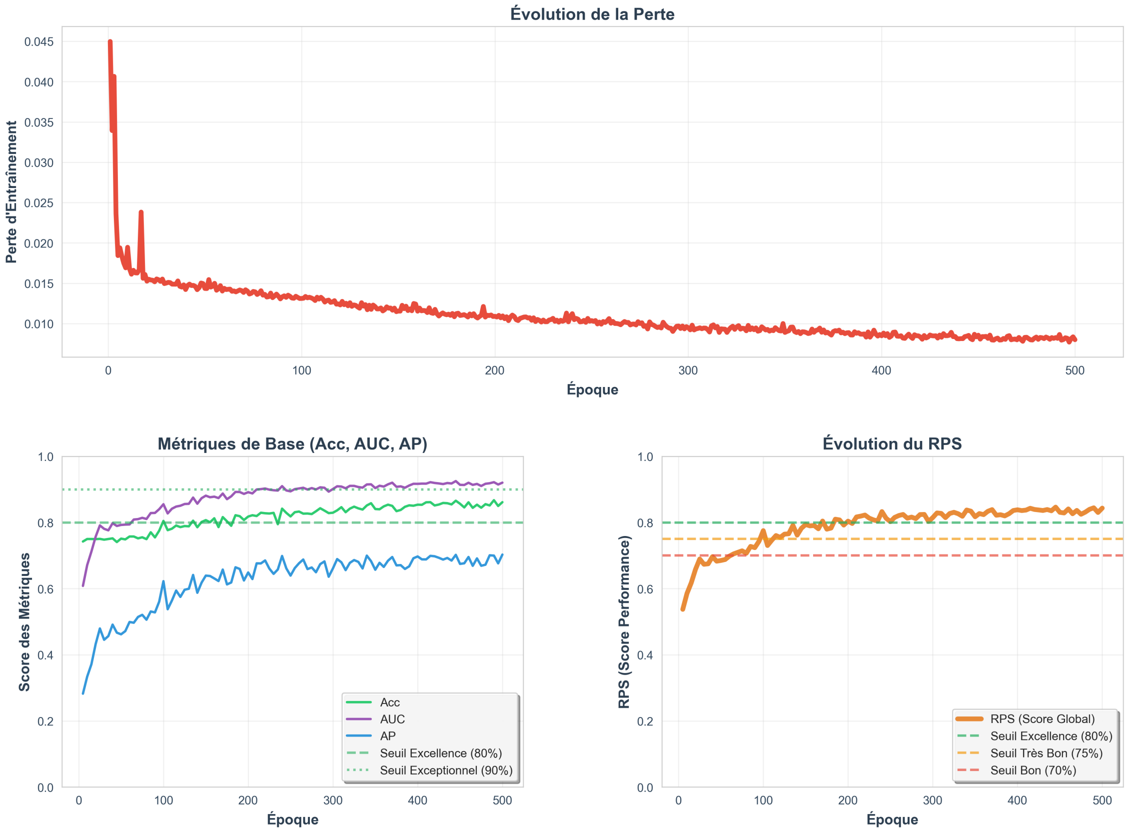


Figure 4.10. Evolution of training metrics for the optimal configuration.

Final Performance

The final performance obtained by the GNN model on the test set is presented in Table 4.4.

Metric	Obtained value (%)
AUC	92.4
Average Precision (AP)	71.8
Accuracy	86.3
RPS (Recommender Performance Score)	**85.0**

Table 4.4. Final performance of the recommended GNN model (test set)

Fine-Tuning Phase

Given the relatively small size of the database, a second training phase was performed on the complete set of available data (without separation into test set), preceded by a *fine-tuning*[4] step. This strategy allowed full exploitation of present information, while benefiting from knowledge acquired during initial supervised training, to generate more representative and robust embeddings for final production use [168].

Algorithm 2: GNN Model Fine-tuning

Require: Pre-trained model: `model`, dataset: `data`

Ensure: Fine-tuned model and associated performance metrics

1: $\texttt{lr} \leftarrow 0.0005$ ▷ Low learning rate for fine weight adjustment

2: $\texttt{optimizer} \leftarrow \textsc{Adam}(\texttt{model}.parameters(), \texttt{lr})$

3: $\texttt{epochs} \leftarrow 20$ ▷ Limited number of epochs to avoid overfitting

4: $\texttt{model}, \texttt{results} \leftarrow \text{TRAIN\_MODEL}(\texttt{model}, \texttt{data}, \texttt{optimizer}, \texttt{epochs})$

5: **return** Fine-tuned model `model` and performance metrics `results`

The performance obtained after this fine-tuning step and complete training is synthesized in Table 4.5.

Metric	**Obtained value (%)**
AUC	94.77
Average Precision (AP)	87.41
Accuracy	87.97
RPS (Recommender Performance Score)	**91.20**

Table 4.5. Final performance of the GNN model after fine-tuning and training on the entire database

A script was developed and constitutes the main evaluation tool for the GNN physiotherapy exercise recommendation model on the complete dataset. This program performs an exhaustive evaluation of recommendation system performance. The evaluation adopts an individualized approach where, for each patient, the model generates a number of recommendations K corresponding exactly to the number of exercises that were actually prescribed to them in the reference data.

Finally, Figure 4.11 presents a visual comparison between the ten most recommended exercises by the model and the ten most frequently present exercises in the original database. This analysis allows evaluating the consistency of suggestions generated by the recommendation system compared to the real distribution of prescribed exercises. Good overlap between these two lists indicates that the model has

[4] *Fine-tuning* refers to a phase of adjusting weights of a previously trained model, on the complete set of data or on a specific subset, with the goal of improving its specialization and final performance. This technique allows preserving learned embeddings while adapting them more finely to the global data distribution.

well captured the global trends in the data, while divergences may reveal learning biases or discovery capabilities for under-represented exercises.

Figure 4.11. Comparison between the ten most recommended exercises by the model and the ten most frequent exercises in the database.

4.7 Search for Optimal MF Model Configuration

For the MF model, hyperparameter optimization is considerably simplified compared to complex neural network architectures, as there is only one critical hyperparameter to adjust: the number of latent factors.

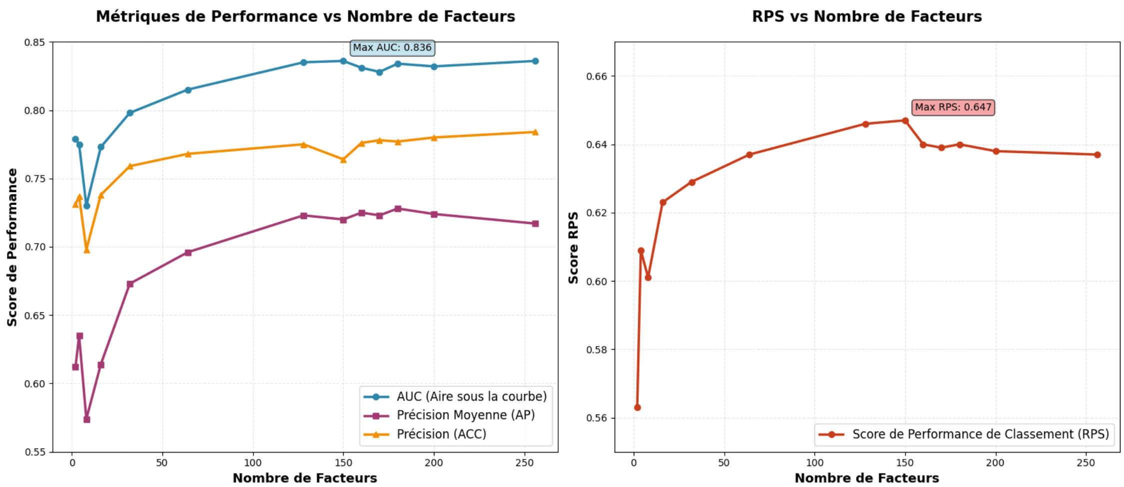


Figure 4.12. Analysis of the impact of the number of latent factors on MF model performance.

Figure 4.12 presents an analysis of the impact of the number of latent factors on MF model performance, testing 13 different configurations ranging from 2 to 256 factors. Basic metrics (left panel) reveal distinct trends: AUC (blue curve) progresses rapidly up to 32 factors then stabilizes around 83-84% with an optimum of 83.6% at 150 factors. Accuracy (orange curve) shows constant improvement, going from 73.1% with 2 factors to 78.4% with 256 factors, with a notable drop at 8 factors (69.8%). Average precision (purple curve) presents more erratic evolution, starting from 61.2% with 2 factors to reach its maximum of 72.8% at 180 factors. RPS score evolution (right panel) highlights the optimal configuration at 150 factors with a score of 64.7%, showing rapid progression up to 128 factors then stabilization. This analysis demonstrates that MF model performance reaches its optimum around 150 factors, with diminishing returns or stagnation beyond this value, suggesting a balance between model complexity and generalization.

4.7.1 Optimal Configuration

The optimal configuration of the matrix factorization (MF) model is summarized in table Table 4.6, accompanied by metrics obtained on the test set. The generated model has a total of 73,085 trainable parameters.

Hyperparameter	**Optimal value**	**AUC (%)**	**AP (%)**	**ACC (%)**	**RPS (%)**
Number of latent factors	150	83.6	72.0	76.4	64.7

Table 4.6. Hyperparameters and performance of the best MF model configuration.

4.8 Graphical Interface

To facilitate the use of both recommendation systems, a web application has been developed. Relatively simple to use, it allows, as illustrated in Figure 4.13, to generate exercise recommendations for patients already present in the database, as well as for new patients who were absent during training, while offering the possibility to choose between the MF or GNN approaches.

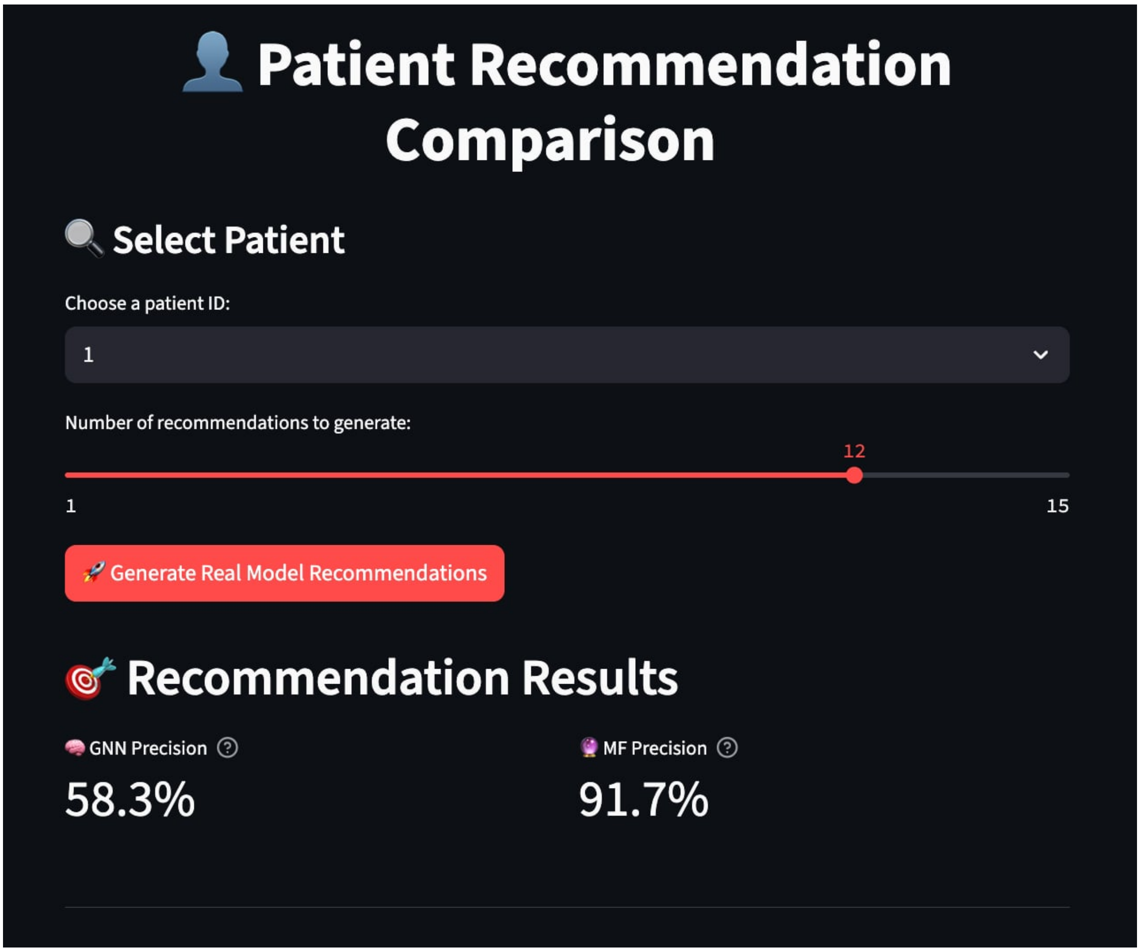


Figure 4.13. Exercise generation for patients present in the database.

Figure 4.14 shows the results of the generated recommendations.

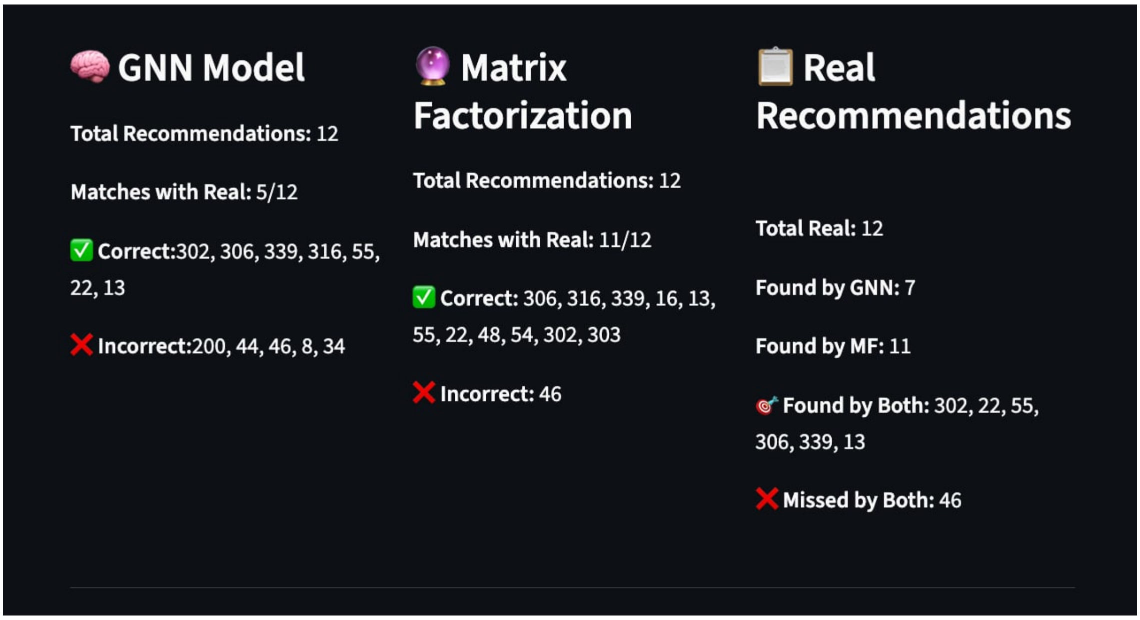


Figure 4.14. Recommendation results.

The application also allows generating recommendations for new patients, as illustrated in Figure 4.15.

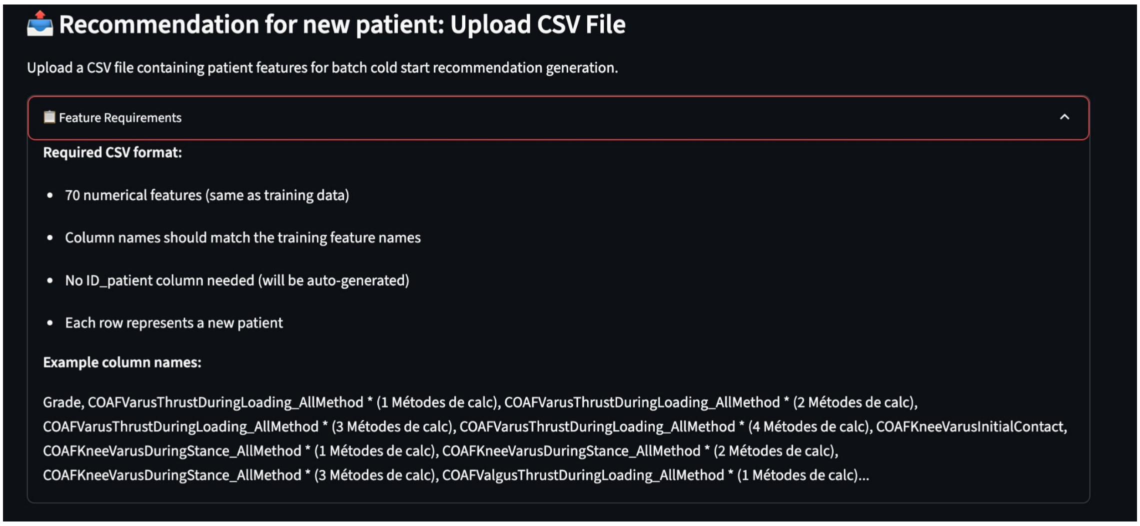


Figure 4.15. Recommendation generation for a new patient (*cold-start* case).

4.9 Performance Comparison between GNN and MF Models

Comparative evaluation between our GNN and MF models reveals manifest superiority of the graph-based architecture. As illustrated in Figure 4.16, the GNN model significantly outperforms the MF model on all evaluation metrics, with an average relative improvement of +22.7%. The most remarkable improvement concerns the RPS score, with a gain of +26.5% (91.20% versus 64.70%).

This superiority is explained by three key factors:

- First, unlike MF which limits itself to historical interactions, GNN explicitly integrates patient and exercise characteristics, allowing capture of complex semantic relationships.
- Second, the attention mechanism via GATs dynamically assigns differentiated weights to relationships, while MF treats all interactions uniformly.
- Third, MF suffers from the cold start problem and generates embeddings that are specific to training data, while GNN learns generalizable logical relationships between pathological profiles and therapeutic properties.

These results demonstrate the fundamental advantage of GNNs in modeling structural dependencies for rehabilitation exercise recommendation, generating more expressive embeddings and clinically more relevant predictions.

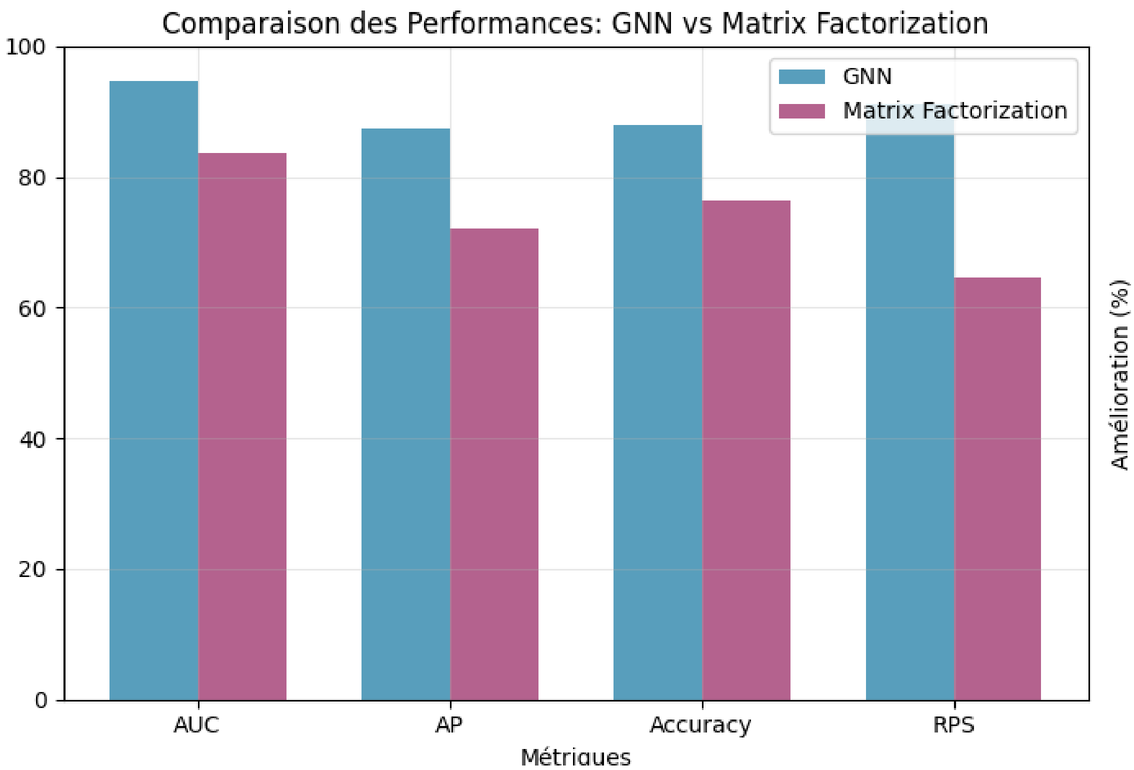


Figure 4.16. Performance comparison between GNN and MF models.

4.10 Conclusion

Experiments demonstrate the superiority of the GNN approach over MF for rehabilitation exercise recommendation. The identified optimal architecture (4 layers, 64 hidden dimensions, 128 embeddings, 2 attention heads) achieves an excellent balance between performance and computational efficiency. Analysis reveals the crucial importance of residual connections (+7.4% RPS) and MLP decoder (+4.3% RPS).

GNN superiority is explained by its ability to exploit intrinsic characteristics of patients and exercises, to dynamically assign weights via attention, and to generalize to new patients, unlike MF limitations. The Embedding Encoder guarantees consistent recommendations for new patients, validating the clinical applicability of the system and opening the way toward optimized therapeutic personalization.

General conclusion

This study has enabled the design, implementation, and evaluation of a therapeutic exercise recommendation system specifically adapted to patients with gonarthrosis, leveraging Graph Neural Networks (GNN) and Matrix Factorization (MF). The experimental results have clearly demonstrated the significant superiority of the GNN-based model compared to the MF approach. The GNN achieved a Recommender Performance Score (RPS) of 91.20

This enhanced performance is explained by the GNN's intrinsic ability to integrate rich characteristics of patients and exercises, to capture complex non-linear relationships via its multilayer architecture, and to dynamically weight the importance of connections through its attention mechanism. In contrast, MF, while less computationally expensive, is limited by its inability to directly incorporate explicit features and by its uniform treatment of interactions, which reduces its capacity to model nuanced patient-exercise dynamics.

An essential contribution of this work lies in the effective resolution of the "cold start" problem through the development and integration of an Embedding Encoder. This component ensures that the system can generate coherent and relevant recommendations for new patients, even in the absence of extensive historical interaction data. By projecting new patient characteristics into the latent space learned by the GNN or MF model, the Embedding Encoder allows the system to provide personalized therapeutic suggestions from the first contact, thus improving its clinical applicability.

The analysis has also highlighted the crucial role of architectural components such as residual connections and the MLP decoder, which have significantly improved performance by solving issues like gradient attenuation and by increasing the model's capacity to learn complex non-linear patterns.

In summary, this research not only validates the potential of GNNs for developing highly effective health recommendation systems, but it also establishes a robust framework for personalized prescription of therapeutic exercises in the context of gonarthrosis. By surpassing generalized exercise programs, the proposed system paves the way for optimizing rehabilitation outcomes, improving patient adherence, and ultimately enhancing the quality of life for individuals affected by this debilitating condition.

Future work could explore the integration of real-time feedback mechanisms and the expansion of the dataset to further refine and validate the system's capabilities in various clinical environments.

Bibliography

[1] Arthrose du genou (gonarthrose), symptômes et traitements, . URL https://www.ramsaysante.fr/vous-etes-patient-en-savoir-plus-sur-ma-pathologie/arthrose-du-genou.

[2] Henry Knipe. Osteoarthritis of the knee | Radiology Reference Article | Radiopaedia.org. https://radiopaedia.org/articles/osteoarthritis-of-the-knee.

[3] What Is Osteoarthritis (OA) of the Knee? https://my.clevelandclinic.org/health/diseases/21750-osteoarthritis-knee, .

[4] Overview: Osteoarthritis of the knee. In *InformedHealth.Org [Internet]*. Institute for Quality and Efficiency in Health Care (IQWiG), May 2024.

[5] Marco Gupton, Onyebuchi Imonugo, Asa C. Black, Marjorie V. Launico, and Robert R. Terreberry. Anatomy, Bony Pelvis and Lower Limb, Knee. In *StatPearls*. StatPearls Publishing. URL http://www.ncbi.nlm.nih.gov/books/NBK500017/.

[6] M. Shahabpour, N. DeMeyere, M. DeMaeseneer, P. David, F. DeRidder, and T. Stadnik. Anatomie normale du genou en imagerie par résonance magnétique. *EMC - Radiologie*, 2(2):165–182, May 2005. ISSN 1762-4185. doi: 10.1016/j.emcrad.2004.12.001.

[7] Turner A Blackburn and Emily Craig. Knee Anatomy: A Brief Review. *Physical Therapy*, 60(12):1556–1560, December 1980. ISSN 1538-6724. doi: 10.1093/ptj/60.12.1556.

[8] Anatomie du genou : os, cartilage, muscles, tendons, ménisques, nerfs, .

[9] Leg skeletal anatomy: MedlinePlus Medical Encyclopedia Image. https://medlineplus.gov/ency/imagepages/8844.htm.

Bibliography

[10] Femur (Thighbone): Anatomy, Function & Common Conditions. https://my.clevelandclinic.org/health/body/22503-femur, .

[11] Femur. https://www.kenhub.com/en/library/anatomy/femur, .

[12] Dr.AFOUTNI A. Muscles de la cuisse, 2022.

[13] Articulation du genou. https://www.kenhub.com/fr/library/anatomie/articulation-du-genou, .

[14] Docteur Philippe Paris Chirurgien Orthopédiste Marseille | Genou et sport, Hanche, Cheville et Pied. https://www.chirortho13.fr/.

[15] The Knee Joint - Articulations - Movements - Injuries - TeachMeAnatomy, . URL https://teachmeanatomy.info/lower-limb/joints/knee-joint/.

[16] Tibia (Shin Bone): Location, Anatomy & Common Conditions. URL https://my.clevelandclinic.org/health/body/23026-tibia.

[17] Perone anatomie artistique. https://medecine-des-arts.com/fr/article/perone-anatomie-artistique-lecon-43.php, .

[18] Péroné - LAROUSSE. https://www.larousse.fr/encyclopedie/medical/p%C3%A9ron%C3%A9/ .

[19] The Patella - Surface Anatomy - Functions - Dislocation - TeachMeAnatomy. https://teachmeanatomy.info/lower-limb/bones/patella/, .

[20] Chandler F. Cox, Margaret A. Sinkler, Asa C. Black, Marjorie V. Launico, and John B. Hubbard. Anatomy, Bony Pelvis and Lower Limb, Knee Patella. In *StatPearls*. StatPearls Publishing, Treasure Island (FL), 2025.

[21] Patella (Rotule). https://www.kenhub.com/fr/library/anatomie/patella-fr, .

[22] Knee Joint: Function & Anatomy. https://my.clevelandclinic.org/health/body/24777-knee-joint, .

[23] Knee Joint Anatomy: Overview, Gross Anatomy, Natural Variants. 2025-02-19-18:09.

[24] damien. Quadriceps Anatomy And Training - Blog Eric Favre UK, July 2021.

[25] Symptômes et diagnostic de la rupture du tendon rotulien | Dr Paillard.

[26] Knee Anatomy: Muscles, Ligaments, and Cartilage. https://www.joionline.net/library/knee-anatomy-muscles-ligaments-and-cartilage/, .

[27] Soigner la maladie d'Osgood-Schlatter - ORTHOREPASS. https://www.orthorepass.com/soigner-la-maladie-dosgood-schlatter/.

[28] Patellar tendinitis - Symptoms & causes - Mayo Clinic. https://www.mayoclinic.org/diseases-conditions/patellar-tendinitis/symptoms-causes/syc-20376113, .

[29] Syed Zahid Hussain Bokhari. Tendonitis: The major cause of pain in osteoarthritis knee joint. *Journal of Ayub Medical College, Abbottabad: JAMC*, 24(3-4):109–112, 2012. ISSN 1025-9589.

[30] Jawad F. Abulhasan and Michael J. Grey. Anatomy and Physiology of Knee Stability. *Journal of Functional Morphology and Kinesiology*, 2(4):34, December 2017. ISSN 2411-5142. doi: 10.3390/jfmk2040034.

[31] Dr. Justin Newman | Orthopedic Surgeon Parker | Sports Medicine Castle Rock CO. https://www.justinnewmanmd.com/normal-anatomy-of-the-knee-joint/.

[32] Joint Instability and developing degenerative joint disease and progressive osteoarthritis.

[33] Patient Education. https://www.concordortho.com/patient-resources/patient-education/topic/8e9ebd3d2fd79fd452e155cb85d1bade, .

[34] James Kevin Bryceland, Andrew John Powell, and Thomas Nunn. Knee Menisci: Structure, Function, and Management of Pathology. 8(2):99–104. ISSN 1947-6035, 1947-6043. doi: 10.1177/1947603516654945. URL https://journals.sagepub.com/doi/10.1177/1947603516654945.

[35] Anatomie du ménisque, . URL https://www.chirurgiedusport.com/nos-specialites/anatomie-du-menisque/.

[36] Knee – Degenerative Meniscal Tears - NHS Ayrshire & Arran, November 2023.

[37] Knee meniscus tear with arthritis: Treatment, link, and more. https://www.medicalnewstoday.com/articles/how-to-treat-knee-meniscus-tear-with-arthritis, October 2022.

[38] Rouhin Sen and John A. Hurley. Osteoarthritis. In *StatPearls*. StatPearls Publishing, Treasure Island (FL), 2025.

[39] Yuchen He, Zhong Li, Peter G. Alexander, Brian D. Ocasio-Nieves, Lauren Yocum, Hang Lin, and Rocky S. Tuan. Pathogenesis of Osteoarthritis: Risk Factors, Regulatory Pathways in Chondrocytes, and Experimental Models. *Biology*, 9(8):194, August 2020. ISSN 2079-7737. doi: 10.3390/biology9080194.

[40] Olivia Carmen Timnea, Ana Maria Stănescu, Larisa Potop, and Andreea Consuela Timnea. Kinetic Rehabilitation For Adults With Gonarthrosis. *European Proceedings of Social and Behavioural Sciences*, Education and Sports Science in the 21st Century, February 2019. ISSN 2357-1330. doi: 10.15405/epsbs.2019.02.46.

[41] L. Sharma, J. Song, D. T. Felson, S. Cahue, E. Shamiyeh, and D. D. Dunlop. The role of knee alignment in disease progression and functional decline in knee osteoarthritis. *JAMA*, 286(2):188–195, July 2001. ISSN 0098-7484. doi: 10.1001/jama.286.2.188.

[42] Knee Osteoarthritis. https://www.physio-pedia.com/Knee_Osteoarthritis, .

[43] Stefano Coaccioli, Piercarlo Sarzi-Puttini, Panagiotis Zis, Giuseppe Rinonapoli, and Giustino Varrassi. Osteoarthritis: New Insight on Its Pathophysiology. 11(20):6013. ISSN 2077-0383. doi: 10.3390/jcm11206013. URL https://www.mdpi.com/2077-0383/11/20/6013.

[44] Anna Litwic, Mark H. Edwards, Elaine M. Dennison, and Cyrus Cooper. Epidemiology and burden of osteoarthritis. *British Medical Bulletin*, 105(1): 185–199, March 2013. ISSN 0007-1420. doi: 10.1093/bmb/lds038.

[45] Youssef Ouakrim. Classification de sujets asymptomatiques et gonarthrosiques en fonction des données cinématiques comparaison de l'approche globale et de l'approche locale. Master's thesis, École de technologie supérieure, Montréal, October 2011.

[46] Knee Kinesiography, a Dynamic Knee Exam | KNEEKG®. https://emovi.ca/, .

[47] What Is a Knee Kinesiography Exam? | EMOVI® | KNEEKG®, .

[48] Julien Clément, Panagiota Toliopoulos, Nicola Hagemeister, François Desmeules, Alexandre Fuentes, and Pascal-André Vendittoli. Healthy 3D knee

kinematics during gait: Differences between women and men, and correlation with x-ray alignment. *Gait & Posture*, 64:198–204, July 2018. ISSN 1879-2219. doi: 10.1016/j.gaitpost.2018.06.024.

[49] Dafina Bytyqi, Bujar Shabani, Sebastien Lustig, Laurence Cheze, Natyra Karahoda Gjurgjeala, and Philippe Neyret. Gait knee kinematic alterations in medial osteoarthritis: Three dimensional assessment. *International Orthopaedics*, 38(6):1191–1198, June 2014. ISSN 1432-5195. doi: 10.1007/s00264-014-2312-3.

[50] Neila Mezghani, Youssef Ouakrim, Alexandre Fuentes, Amar Mitiche, Nicola Hagemeister, Pascal-André Vendittoli, and Jacques A. de Guise. Mechanical biomarkers of medial compartment knee osteoarthritis diagnosis and severity grading: Discovery phase. *Journal of Biomechanics*, 52:106–112, February 2017. ISSN 1873-2380. doi: 10.1016/j.jbiomech.2016.12.022.

[51] Neila Mezghani, Karine Boivin, Katia Turcot, Rachid Aissaoui, Nicola Hagmeister, and Jacques A. De Guise. Hierarchical analysis and classification of asymptomatic and knee osteoarthritis gait patterns using a wavelet representation of kinetic data and the nearest neighbor classifier. *Journal of Mechanics in Medicine and Biology*, 08(01):45–54, March 2008. ISSN 0219-5194. doi: 10.1142/S0219519408002474.

[52] Xiaolong Zeng, Limin Ma, Zefeng Lin, Wenhan Huang, Zhiqiang Huang, Yu Zhang, and Chuanbin Mao. Relationship between Kellgren-Lawrence score and 3D kinematic gait analysis of patients with medial knee osteoarthritis using a new gait system. *Scientific Reports*, 7(1):4080, June 2017. ISSN 2045-2322. doi: 10.1038/s41598-017-04390-5.

[53] David J. Hunter. Focusing osteoarthritis management on modifiable risk factors and future therapeutic prospects. *Therapeutic Advances in Musculoskeletal Disease*, 1(1):35–47, October 2009. ISSN 1759-720X. doi: 10.1177/1759720X09342132.

[54] N. Mezghani, Y. Ouakrim, A. Fuentes, N. Hagemeister, R. Aissaoui, M. Pelletier, and J. De Guise. Knee osteoarthritis severity assessment using knee kinematic data classification. 20:S97. ISSN 10634584. doi: 10.1016/j.joca.2012.02.102. URL https://linkinghub.elsevier.com/retrieve/pii/S1063458412001744.

[55] Nathaly Gaudreault, Neila Mezghani, Katia Turcot, Nicola Hagemeister, Karine Boivin, and Jacques A. de Guise. Effects of physiotherapy treatment on knee osteoarthritis gait data using principal component analysis. *Clinical Biomechanics (Bristol, Avon)*, 26(3):284–291, March 2011. ISSN 1879-1271. doi: 10.1016/j.clinbiomech.2010.10.004.

[56] Neila Mezghani, Sabine Husse, Karine Boivin, Katia Turcot, Rachid Aissaoui, Nicola Hagemeister, and Jacques A. de Guise. Automatic classification of asymptomatic and osteoarthritis knee gait patterns using kinematic data features and the nearest neighbor classifier. *IEEE transactions on biomedical engineering*, 55(3):1230–1232, March 2008. ISSN 0018-9294. doi: 10.1109/TBME.2007.905388.

[57] Automatic classification of 3D kinetic data of healthy and ACL deficient participants. In *ResearchGate*.

[58] M.J. Beynon, L. Jones, and C.A. Holt. Classification of osteoarthritic and normal knee function using three-dimensional motion analysis and the Dempster-Shafer theory of evidence. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 36(1):173–186, January 2006. ISSN 1558-2426. doi: 10.1109/TSMCA.2006.859098.

[59] Phases of the normal gait cycle | Download Scientific Diagram. https://www.researchgate.net/figure/Phases-of-the-normal-gait-cycle_fig3_309362425.

[60] Figure 1. Directions of displacement. AP: Anterior-posterior direction... https://www.researchgate.net/figure/Directions-of-displacement-AP-Anterior-posterior-direction-forward-backward_fig6_225059668, .

[61] Neila Mezghani, Imene Mechmeche, Amar Mitiche, Youssef Ouakrim, and Jacques A. de Guise. An analysis of 3D knee kinematic data complexity in knee osteoarthritis and asymptomatic controls. *PloS One*, 13(10):e0202348, 2018. ISSN 1932-6203. doi: 10.1371/journal.pone.0202348.

[62] Sébastien Lustig, Robert A. Magnussen, Laurence Cheze, and Philippe Neyret. The KneeKG system: A review of the literature. *Knee Surgery, Sports Traumatology, Arthroscopy*, 20(4):633–638, April 2012. ISSN 1433-7347. doi: 10.1007/s00167-011-1867-4.

[63] What is a Recommendation System? | Data Science | NVIDIA Glossary. https://www.nvidia.com/en-us/glossary/recommendation-system/, .

[64] Prem Melville and Vikas Sindhwani. Recommender Systems. In Claude Sammut and Geoffrey I. Webb, editors, *Encyclopedia of Machine Learning*, pages 829–838. Springer US, Boston, MA, 2010. ISBN 978-0-387-30164-8. doi: 10.1007/978-0-387-30164-8_705.

[65] Linyuan Lü, Matúš Medo, Chi Ho Yeung, Yi-Cheng Zhang, Zi-Ke Zhang, and Tao Zhou. Recommender systems. *Physics Reports*, 519(1):1–49, October 2012. ISSN 0370-1573. doi: 10.1016/j.physrep.2012.02.006.

[66] Francesco Ricci, Lior Rokach, and Bracha Shapira. Introduction to Recommender Systems Handbook. In Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor, editors, *Recommender Systems Handbook*, pages 1–35. Springer US, Boston, MA, 2011. ISBN 978-0-387-85820-3. doi: 10.1007/978-0-387-85820-3_1.

[67] Peter Brusilovsky and Eva Millán. User Models for Adaptive Hypermedia and Adaptive Educational Systems. In Peter Brusilovsky, Alfred Kobsa, and Wolfgang Nejdl, editors, *The Adaptive Web: Methods and Strategies of Web Personalization*, pages 3–53. Springer, Berlin, Heidelberg, 2007. ISBN 978-3-540-72079-9. doi: 10.1007/978-3-540-72079-9_1.

[68] Tariq Mahmood and Francesco Ricci. Improving recommender systems with adaptive conversational strategies. In *Proceedings of the 20th ACM Conference on Hypertext and Hypermedia*, HT '09, pages 73–82, New York, NY, USA, June 2009. Association for Computing Machinery. ISBN 978-1-60558-486-7. doi: 10.1145/1557914.1557930.

[69] Paul Resnick and Hal R. Varian. Recommender systems. *Commun. ACM*, 40 (3):56–58, March 1997. ISSN 0001-0782. doi: 10.1145/245108.245121.

[70] Joeran Beel, Bela Gipp, Stefan Langer, and Corinna Breitinger. Research-paper recommender systems: A literature survey. *International Journal on Digital Libraries*, 17(4):305–338, November 2016. ISSN 1432-1300. doi: 10.1007/s00799-015-0156-0.

[71] Types of recommender system. https://www.researchgate.net/figure/Types-of-recommender-system_fig2_359335702.

[72] Erion Çano and Maurizio Morisio. Hybrid Recommender Systems: A Systematic Literature Review. *Intelligent Data Analysis*, 21(6):1487–1524, November 2017. ISSN 1088467X, 15714128. doi: 10.3233/IDA-163209.

[73] christine. Les systèmes de recommandation : une catégorisation, September 2018.

[74] Luigi Palopoli, Domenico Rosaci, and Giuseppe M. L. Sarné. A Multi-tiered Recommender System Architecture for Supporting E-Commerce. In Giancarlo Fortino, Costin Badica, Michele Malgeri, and Rainer Unland, editors, *Intelligent Distributed Computing VI*, pages 71–81, Berlin, Heidelberg, 2013. Springer. ISBN 978-3-642-32524-3. doi: 10.1007/978-3-642-32524-3_10.

[75] R Dong and L Tokarchuk. Digging Friendship: Paper Recommendation in Social Network.

[76] Blerina Lika, Kostas Kolomvatsos, and Stathes Hadjiefthymiades. Facing the cold start problem in recommender systems. 41(4):2065–2073. ISSN 09574174. doi: 10.1016/j.eswa.2013.09.005. URL https://linkinghub.elsevier.com/retrieve/pii/S0957417413007240.

[77] Deepak Kumar Panda and Sanjog Ray. Approaches and algorithms to mitigate cold start problems in recommender systems: A systematic literature review. *Journal of Intelligent Information Systems*, 59(2):341–366, October 2022. ISSN 1573-7675. doi: 10.1007/s10844-022-00698-5.

[78] Rabaa Alabdulrahman and Herna Viktor. Catering for unique tastes: Targeting grey-sheep users recommender systems through one-class machine learning. *Expert Systems with Applications*, 166:114061, March 2021. ISSN 0957-4174. doi: 10.1016/j.eswa.2020.114061.

[79] Paul. Knowledge-Based Recommenders. URL https://medium.com/@paul0/knowledge-based-rs-b4547b84108a.

[80] Rakhi Seth and Aakanksha Sharaff. A Comparative Overview of Hybrid Recommender Systems: Review, Challenges, and Prospects. In Rohit Raja, Kapil Kumar Nagwanshi, Sandeep Kumar, and K. Ramya Laxmi, editors, *Data Mining and Machine Learning Applications*, pages 57–98. Wiley, 1 edition, February 2022. ISBN 978-1-119-79178-2 978-1-119-79252-9. doi: 10.1002/9781119792529.ch3.

[81] Marko Balabanović and Yoav Shoham. Fab: Content-based, collaborative recommendation. *Commun. ACM*, 40(3):66–72, March 1997. ISSN 0001-0782. doi: 10.1145/245108.245124.

[82] Lili Wu, Sam Shah, Sean Choi, Mitul Tiwari, and C. Posse. The Browsemaps: Collaborative Filtering at LinkedIn. In *RSWeb@RecSys*, 2014.

[83] Sciforce. Inside recommendations: How a recommender system recommends, February 2020.

[84] Thi Ngoc Trang Tran, Alexander Felfernig, Christoph Trattner, and Andreas Holzinger. Recommender systems in the healthcare domain: State-of-the-art and research issues. *Journal of Intelligent Information Systems*, 57(1):171–201, August 2021. ISSN 1573-7675. doi: 10.1007/s10844-020-00633-6.

[85] Benjamin Stark, Constanze Knahl, Mert Aydin, and Karim Elish. A Literature Review on Medicine Recommender Systems. 10(8). ISSN 21565570, 2158107X. doi: 10.14569/IJACSA.2019.0100802. URL http://thesai.org/Publications/ViewPaper?Volume=10&Issue=8&Code=IJACSA&SerialNo=2.

[86] IBM Watson. https://www.ibm.com/watson, July 2024.

[87] Sai kiran Adlluru. The Rise and Fall of IBM Watson: Lessons from AI's Journey in Healthcare, August 2024.

[88] Rajani Shankar Sadasivam, Sarah L. Cutrona, Rebecca L. Kinney, Benjamin M. Marlin, Kathleen M. Mazor, Stephenie C. Lemon, and Thomas K. Houston. Collective-Intelligence Recommender Systems: Advancing Computer Tailoring for Health Behavior Change Into the 21st Century. *Journal of Medical Internet Research*, 18(3):e4448, March 2016. doi: 10.2196/jmir.4448.

[89] Zheyu Wang, Haoce Huang, Liping Cui, Juan Chen, Jiye An, Huilong Duan, Huiqing Ge, and Ning Deng. Using Natural Language Processing Techniques to Provide Personalized Educational Materials for Chronic Disease Patients in China: Development and Assessment of a Knowledge-Based Health Recommender System. 8(4):e17642. doi: 10.2196/17642. URL https://medinform.jmir.org/2020/4/e17642.

[90] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. 17(6):734–749. ISSN 1041-4347. doi: 10.1109/TKDE.2005.99. URL http://ieeexplore.ieee.org/document/1423975/.

[91] Yao Cai, Fei Yu, Manish Kumar, Roderick Gladney, and Javed Mostafa. Health Recommender Systems Development, Usage, and Evaluation from 2010

to 2022: A Scoping Review. *International Journal of Environmental Research and Public Health*, 19(22):15115, November 2022. ISSN 1660-4601. doi: 10.3390/ijerph192215115.

[92] Mehdi Elahi, Francesco Ricci, and Neil Rubens. A survey of active learning in collaborative filtering recommender systems. *Computer Science Review*, 20: 29–50, May 2016. ISSN 1574-0137. doi: 10.1016/j.cosrev.2016.05.002.

[93] Michael D. Ekstrand. Collaborative Filtering Recommender Systems. *Foundations and Trends® in Human–Computer Interaction*, 4(2):81–173, 2011. ISSN 1551-3955, 1551-3963. doi: 10.1561/1100000009.

[94] Charu C. Aggarwal. *Recommender Systems*. Springer International Publishing, Cham, 2016. ISBN 978-3-319-29657-9 978-3-319-29659-3. doi: 10.1007/978-3-319-29659-3.

[95] J. Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen. Collaborative Filtering Recommender Systems. In Peter Brusilovsky, Alfred Kobsa, and Wolfgang Nejdl, editors, *The Adaptive Web*, volume 4321, pages 291–324. Springer Berlin Heidelberg. ISBN 978-3-540-72078-2. doi: 10.1007/978-3-540-72079-9_9. URL http://link.springer.com/10.1007/978-3-540-72079-9_9.

[96] What is collaborative filtering? | IBM. https://www.ibm.com/think/topics/collaborative-filtering, March 2024.

[97] Example of user-item ratings matrix in a movie recommendation scenario. | Download Scientific Diagram. https://www.researchgate.net/figure/Example-of-user-item-ratings-matrix-in-a-movie-recommendation-scenario_fig2_300646445.

[98] Ms. Tejashri Sharad Phalle and Prof. Shivendu Bhushan. Content Based Filtering And Collaborative Filtering: A Comparative Study. pages 96–100. doi: 10.53555/jaz.v45iS4.4158. URL https://jazindia.com/index.php/jaz/article/view/4158.

[99] The Significance of Collaborative Filtering in Recommender Systems. https://www.cloudthat.com/resources/blog/the-significance-of-collaborative-filtering-in-recommender-systems.

[100] What Is Collaborative Filtering: A Simple Introduction | Built In. https://builtin.com/data-science/collaborative-filtering-recommender-system, .

[101] User-based and Item-based Collaborative Filtering | Download Scientific Diagram. https://www.researchgate.net/figure/User-based-and-Item-based-Collaborative-Filtering_fig2_366902172.

[102] Mukund Deshpande and George Karypis. Item-based top- N recommendation algorithms. *ACM Transactions on Information Systems*, 22(1):143–177, January 2004. ISSN 1046-8188, 1558-2868. doi: 10.1145/963770.963776.

[103] Wenbin Yue, Zidong Wang, Bo Tian, Mark Pook, and Xiaohui Liu. A Hybrid Model- and Memory-Based Collaborative Filtering Algorithm for Baseline Data Prediction of Friedreich's Ataxia Patients. *IEEE Transactions on Industrial Informatics*, 17(2):1428–1437, February 2021. ISSN 1551-3203, 1941-0050. doi: 10.1109/TII.2020.2984540.

[104] Hael Al-Bashiri, Mansoor Abdullateef Abdulgabber, Awanis Romli, and Fadhl Hujainah. Collaborative Filtering Recommender System: Overview and Challenges. *Advanced Science Letters*, 23(9):9045–9049, September 2017. ISSN 1936-6612. doi: 10.1166/asl.2017.10020.

[105] Muffaddal Qutbuddin. Comprehensive Guide on Item Based Recommendation Systems, February 2024.

[106] Norm Matloff. Collaborative Filtering in Recommender Systems: A Short Introduction.

[107] Abhaya Kumar Sahoo, Chittaranjan Pradhan, Rabindra Kumar Barik, and Harishchandra Dubey. DeepReco: Deep Learning Based Health Recommender System Using Collaborative Filtering. *Computation*, 7(2):25, May 2019. ISSN 2079-3197. doi: 10.3390/computation7020025.

[108] Dheeraj Bokde, Sheetal Girase, and Debajyoti Mukhopadhyay. Matrix Factorization Model in Collaborative Filtering Algorithms: A Survey. *Procedia Computer Science*, 49:136–146, 2015. ISSN 18770509. doi: 10.1016/j.procs.2015.04.237.

[109] Implementing Matrix Factorization Technique for Recommender Systems from scratch | by Sadak Vali | Medium. https://medium.com/@rebirth4vali/implementing-matrix-factorization-technique-for-recommender-systems-from-scratch-7828c9166d3c.

[110] Rishabh Ahuja, Arun Solanki, and Anand Nayyar. Movie Recommender System Using K-Means Clustering AND K-Nearest Neighbor. In *2019 9th Inter-*

national Conference on Cloud Computing, Data Science & Engineering (Confluence), pages 263–268, Noida, India, January 2019. IEEE. ISBN 978-1-5386-5933-5. doi: 10.1109/CONFLUENCE.2019.8776969.

[111] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, pages 285–295, Hong Kong Hong Kong, April 2001. ACM. ISBN 978-1-58113-348-6. doi: 10.1145/371920.372071.

[112] Clustering basics and a demonstration in clustering infrastructure pathways – Water Programming: A Collaborative Research Blog. https://waterprogramming.wordpress.com/2022/03/16/clustering-basics-and-a-demonstration-in-clustering-infrastructure-pathways/.

[113] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural Collaborative Filtering, August 2017.

[114] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web*, pages 173–182, Perth Australia, April 2017. International World Wide Web Conferences Steering Committee. ISBN 978-1-4503-4913-0. doi: 10.1145/3038912.3052569.

[115] ZEMIM. Deep Neural Network (DNN) Explained, April 2024.

[116] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph Neural Networks in Recommender Systems: A Survey, April 2022.

[117] Sergios Karagiannakos. Graph Neural Networks - An overview. https://theaisummer.com/Graph_Neural_Networks/, February 2020.

[118] Carmel Wenga, Majirus Fansi, Sébastien Chabrier, Jean-Martial Mari, and Alban Gabillon. A Comprehensive Review on Non-Neural Networks Collaborative Filtering Recommendation Systems, 2021.

[119] Wenbin Wu, Zhanyong Qi, Jiawei Tian, Bixi Wang, Minyi Tang, and Xuan Liu. An Enhanced Latent Factor Recommendation Approach for Sparse Datasets of E-Commerce Platforms. *Systems*, 13(5):372, May 2025. ISSN 2079-8954. doi: 10.3390/systems13050372.

[120] Aleksandra Pawlicka, Marek Pawlicki, Rafał Kozik, and Ryszard S. Choraś. A Systematic Review of Recommender Systems and Their Applications in Cybersecurity. 21(15):5248. ISSN 1424-8220. doi: 10.3390/s21155248. URL https://www.mdpi.com/1424-8220/21/15/5248.

[121] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph Neural Networks in Recommender Systems: A Survey. *ACM Computing Surveys*, 55 (5):1–37, May 2023. ISSN 0360-0300, 1557-7341. doi: 10.1145/3535101.

[122] Rianne van den Berg, Thomas N. Kipf, and Max Welling. Graph Convolutional Matrix Completion, October 2017.

[123] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 974–983, July 2018. doi: 10.1145/3219819.3219890.

[124] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation, July 2020.

[125] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. Neural Graph Collaborative Filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 165–174, Paris France, July 2019. ACM. ISBN 978-1-4503-6172-9. doi: 10.1145/3331184.3331267.

[126] Annielle Mendes Brito Da Silva, Natiele Carla Da Silva Ferreira, Luiza Amara Maciel Braga, Fabio Batista Mota, Victor Maricato, and Luiz Anastacio Alves. Graph Neural Networks: A Bibliometric Mapping of the Research Landscape and Applications. *Information*, 15(10):626, October 2024. ISSN 2078-2489. doi: 10.3390/info15100626.

[127] Applications of Graph Data Structure | GeeksforGeeks. https://www.geeksforgeeks.org/applications-of-graph-data-structure/.

[128] Représentation graphique de (a.) un réseau non-dirigé et... https://www.researchgate.net/figure/representation-graphique-de-a-un-reseau-non-dirige-et-non-pondere-a-trois-noeuds-et_fig4_277587050.

[129] Variant of graphs found in nature and science. Directed VS Undirected... https://www.researchgate.net/figure/Variant-of-graphs-found-in-nature-and-science-Directed-VS-Undirected-Graphs-Static-VS_fig1_365692581.

[130] What is a Hypergraph? – Be on the Right Side of Change, April 2022.

[131] What is a Graph Neural Network | IBM. https://www.ibm.com/think/topics/graph-neural-network, February 2025.

[132] James H. Tanis, Chris Giannella, and Adrian V. Mariano. Introduction to Graph Neural Networks: A Starting Point for Machine Learning Engineers, 2024.

[133] DiploDoc. Les réseaux de neurones en graphes (GNN) permettent d'étudier les interactions entre les. . . , May 2021.

[134] Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. GNNExplainer: Generating Explanations for Graph Neural Networks. *Advances in Neural Information Processing Systems*, 32:9240–9251, December 2019. ISSN 1049-5258.

[135] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A Neural Probabilistic Language Model. page 19, 2003.

[136] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space, 2013.

[137] Embeddings: Meaning, Examples and How To Compute. https://arize.com/blog-course/embeddings-meaning-examples-and-how-to-compute/.

[138] Graph neural network. *Wikipedia*, May 2025.

[139] Nico Klingler. Graph Neural Networks (GNNs) - Comprehensive Guide, February 2024.

[140] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[141] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated Graph Sequence Neural Networks, September 2017.

[142] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks, February 2018.

[143] Amal Menzli. Graph Neural Network and Some of GNN Applications. URL https://neptune.ai/blog/graph-neural-network-and-some-of-gnn-applications.

[144] Graph Neural Network – Message Passing (GCN) – 1.1 – Denken, November 2022.

[145] What are Graph Neural Networks? https://www.geeksforgeeks.org/what-are-graph-neural-networks/, 2023.

[146] Rudina Seseri. AI Atlas #18: Graph Neural Networks (GNNs). URL https://glasswing.vc/blog/ai-atlas-18-graph-neural-networks-gnns/.

[147] Figure 1. An example User-Item Bipartite Graph with ratings. https://www.researchgate.net/figure/An-example-User-Item-Bipartite-Graph-with-ratings_fig1_361591389, .

[148] Steffen Rendle, Walid Krichene, Li Zhang, and John Anderson. Neural Collaborative Filtering vs. Matrix Factorization Revisited, 2020.

[149] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. A Comprehensive Survey on Graph Neural Networks. 2019. doi: 10.48550/ARXIV.1901.00596.

[150] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph Neural Networks: A Review of Methods and Applications, 2018.

[151] Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks, 2016.

[152] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated Graph Sequence Neural Networks. URL https://arxiv.org/abs/1511.05493.

[153] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph Neural Networks. URL https://arxiv.org/abs/1809.09401.

[154] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. URL https://arxiv.org/abs/1710.10903.

[155] Robin De Croon, Leen Van Houdt, Nyi Nyi Htun, Gregor Štiglic, Vero Vanden Abeele, and Katrien Verbert. Health Recommender Systems: Systematic Review. *Journal of Medical Internet Research*, 23(6):e18035, June 2021. ISSN 1438-8871. doi: 10.2196/18035.

[156] P. Chinnasamy, Wing-Keung Wong, A. Ambeth Raja, Osamah Ibrahim Khalaf, Ajmeera Kiran, and J. Chinna Babu. Health Recommendation System using Deep Learning-based Collaborative Filtering. *Heliyon*, 9(12):e22844, December 2023. ISSN 2405-8440. doi: 10.1016/j.heliyon.2023.e22844.

[157] Su-Cheng Haw, Jayapradha Jayaram, Elham Abdulwahab Anaam, and Heru Agus Santoso. Exploring Recommender Systems in the Healthcare: A Review on Methods, Applications and Evaluations. *International Journal on Robotics, Automation and Sciences*, 6(2):6–15, September 2024. ISSN 2682-860X. doi: 10.33093/ijoras.2024.6.2.2.

[158] Arash Mahyari and Peter Pirolli. Physical Exercise Recommendation and Success Prediction Using Interconnected Recurrent Neural Networks. In *2021 IEEE International Conference on Digital Health (ICDH)*, pages 148–153, September 2021. doi: 10.1109/ICDH52753.2021.00027.

[159] Rob Edge and Kelly Farrah. *Exercise for the Management of Knee Osteoarthritis: A Review of Clinical Effectiveness*. CADTH Rapid Response Reports. Canadian Agency for Drugs and Technologies in Health, Ottawa (ON), 2017.

[160] Waidah Ismail, Ismail Ahmed Al-Qasem Al-Hadi, Crina Grosan, and Rimuljo Hendradi. Improving patient rehabilitation performance in exercise games using collaborative filtering approach. *PeerJ Computer Science*, 7:e599, July 2021. ISSN 2376-5992. doi: 10.7717/peerj-cs.599.

[161] Siyeon Kim, Michael R. Kosorok, Liubov Arbeeva, Todd A. Schwartz, Leigh F. Callahan, Yvonne M. Golightly, Amanda E. Nelson, and Kelli D. Allen. Precision Medicine–Based Machine Learning Analyses to Explore Optimal Exercise Therapies for Individuals With Knee Osteoarthritis: Random Forest-Informed Tree-Based Learning. *The Journal of Rheumatology*, 50(10):1341–1345, October 2023. ISSN 0315-162X, 1499-2752. doi: 10.3899/jrheum.2022-1039.

[162] C. Kokkotis, S. Moustakidis, E. Papageorgiou, G. Giakas, and D. E. Tsaopoulos. Machine learning in knee osteoarthritis: A review. *Osteoarthritis and Cartilage Open*, 2(3):100069, September 2020. ISSN 2665-9131. doi: 10.1016/j.ocarto.2020.100069.

[163] Valerie Dieter, Pia Janssen, and Inga Krauss. Efficacy of the mHealth-Based Exercise Intervention re.flex for Patients With Knee Osteoarthritis: Pilot Randomized Controlled Trial. *JMIR mHealth and uHealth*, 12:e54356, September 2024. ISSN 2291-5222. doi: 10.2196/54356.

[164] Graph machine learning with missing node features. URL https://blog.x.com/engineering/en_us/topics/insights/2022/graph-machine-learning-with-missing-node-features.

[165] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. URL https://arxiv.org/abs/1502.03167.

[166] Crypto1. Gradient Descent Algorithm: How Does it Work in Machine Learning? URL https://www.analyticsvidhya.com/blog/2020/10/how-does-the-gradient-descent-algorithm-work-in-machine-learning/.

[167] Priyansh Soni. ROC-AUC Analysis - A Deep Dive. URL https://www.blog.trainindata.com/auc-roc-analysis/.

[168] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

Appendix A

List of Fields and Explanations

1.1 Orthopedic Characteristics at Time of Prescription

Certain variables in the database describe the orthopedic devices worn by patients at the time of testing or when exercises are prescribed. This information allows better contextualization of the formulated recommendations. The variables concerned are as follows:

- `orthese_stabilisatrice`
- `orthese_plantaire`
- `chaussures_orthopédiques`
- `1ere_ligne_orthese_stab`
- `2eme_ligne_orthese_stab`

These fields indicate the presence or absence of an orthopedic device prescribed as part of functional rehabilitation. The possible values are:

- `1`: device used,
- `0`: device not used,
- `NaN` (empty): information not provided or not documented.

Another variable, `si_oui_genouillere`, indicates the identifier or type of knee brace worn by the patient, if applicable.

It is important to note that systematic imputation of the value `0` for missing fields could lead to misinterpretation. Indeed, an absent value does not necessarily mean that the patient was not wearing a device. It is therefore recommended to:

- exclude patients with missing values in these variables if they are used in the analysis,
- or ignore them if these fields are not exploited in the model.

1.2 Functional Tests Related to Exercise Effects

Certain fields in the database provide results from functional tests evaluating the biomechanical state of the knee and lower limbs. These tests allow measurement of the potential effect of prescribed exercises, by providing indicators of performance, joint mobility, muscle strength, flexibility, and balance.

The main variables are:

- `flow_test`: global performance test, often associated with functional endurance.
- `diff_circ_interligne`: difference in joint circumference, indicator of presence of edema or inflammation.
- `amp_flexion_passive`: passive knee flexion range of motion.
- `amp_extension_passive`: passive knee extension range of motion.
- `force_flexion_genou`: muscle strength in knee flexion.
- `force_extension_genou`: muscle strength in knee extension.
- `force_flexion_hanche`: muscle strength in hip flexion.
- `force_extension_hanche`: muscle strength in hip extension.
- `force_abd_hanche`: hip abduction strength.
- `force_add_hanche`: hip adduction strength.
- `force_rot_INT_hanche`: hip internal rotation strength.
- `force_rot_EXT_hanche`: hip external rotation strength.
- `force_plantaire_cheville`: ankle plantar flexion strength.

Appendix A. List of Fields and Explanations

- `force_dorsale_cheville`: ankle dorsal flexion strength.
- `flexibilite_ischio`: hamstring flexibility.
- `flexibilite_quad_psoas`: quadriceps and psoas flexibility.
- `flexibilite_BIT`: biceps femoris (BIT) flexibility.
- `flexibilite_fessiers_pyramidal`: gluteal and piriformis flexibility.
- `equilibre_unipodal`: single-leg balance assessment.
- `30s_CST`: Chair Stand Test, measuring the number of stands in 30 seconds, indicator of lower limb functional strength.

These measurements offer a multidimensional evaluation of the patient's functional state and can be used as clinical targets, progression indicators, or as explanatory variables in the recommendation model.