Università degli Studi di Padova

DIPARTIMENTO DI MATEMATICA
Master of Science in Data Science

Development of a Recommender System for Adaptive e-Learning

Candidate:
Paolo Prenassi
Matricola 1182874

Supervisor:
Alessandro Sperduti

Thesis submitted in 2020

Contents

Abstract

Recent years have seen the web as an essential mean for Learning and Education, thanks to the almost infinite amount of information shared and to the explosion in development and adoption of e-Learning platforms that allow people to study any topic without the barriers of time, geography and physical participation. In addition to traditional learning content, online platforms allow user-centered approaches, creating an interactive and consequently very effective learning environment. The objective of the thesis is to develop an adaptive learning system for an Italian e-Learning Platform, leader on the market, being able to recommend an optimized learning path for each user. The developed system will be based on machine learning algorithms, which will learn from users' performance and learning characteristics – e.g. time spent learning a single topic, speed of improvement and learning abilities, test scores and completion times – in order to drive the user toward the next best new topic to study or the review on the most appropriate past topics to fill his/her knowledge gap.

The focus will be mainly on mathematics courses, which are, currently, the most requested on the platform.

Introduction

This thesis is intended to illustrate the functioning of the Adaptive e-Learning System developed during the project “Cassandra” for StarRock S.r.l., owner of the registered trademark Redooc as well as the website (www.redooc.com) and its contents (hereinafter referred to as ”Redooc platform”). The Redooc platform has provided online education services since 2014, offering lessons and structured exercises as well as other auxiliary services to support teaching.

The project originates from the idea of demonstrating the feasibility and the benefits of an Adaptive e-Learning System. Therefore a collaboration with Redooc started in order to develop an innovative software solution in the field of adaptive learning, which will be presented in the following chapters.

Specifically, the following tasks have been carried out:

- Analysis of lesson and exercise content, in order to create a knowledge structure based on a network of base concepts;
- Monitoring of users' behaviors, in order to create Machine Learning models able to represent their actual level of knowledge of each concept;
- Recommendation of the optimal learning path for each user according to his/her preferences, goals, actual knowledge, learning speed and needs.

This document is structured as follows. In the first section we introduce the data used as input and the analyses carried out. In the second section we describe in general the functioning and the building blocks of an Adaptive e-Learning System. In the following sections we describe the techniques and algorithms utilized in the development of such systems. Starting from the structuring of the educational contents, then we illustrate the users modelling techniques utilized during the development of Cassandra, and, finally we describe the recommendation techniques adopted.

Chapter 1

Data

In the perspective of developing predictive models with Machine Learning methodology, it is necessary to collect and manipulate historical data. By combining different data sources, using statistics, and leveraging the computational power of machines, it is possible to extract meaningful insights from the data. In fact, Machine Learning algorithms are based on mathematical models and have the ability to automatically learn hidden patterns and improve from experience.

Redooc is a platform that offers educational contents that follows the program of study of the Italian School System starting from the primary school program to the end of the high school program. In particular, it is more focused on the scientific subjects of the High School Program.

Given the greater amount of data present on the Redooc platform, we focused on the material and exams regarding the High School students. Then, according to the indications of the platform owner, we have taken into consideration only the lessons and student activities related to the topics of "Arithmetic and Algebra" ("Aritmetica e Algebra"), "Geometry" ("Geometria"), "Relations and Functions" ("Relazioni e Funzioni") and "Physics" ("Fisica"). Moreover, the initial corpus of texts was enriched utilizing Wikipedia pages (hereinafter referred to as "Wikipedia data") in order to obtain a larger dataset of scientific texts.

1.1 Lesson Material

The material on the platform reflects the study program of the Italian school system and is organized in a hierarchical structure. Each level is characterized by a table with the necessary attributes to understand the relations and the arrangement of the platform contents.

- **Grade** This is the most generic level and divides the platform into the main study cycles of the Italian education system (primary school, secondary school, high school and university). Each grade is divided into the topics taught in each cycle.
- **Topic** It represents the subject, like “Arithmetic and Algebra”, “Geometry”, “Relations and Functions”, and “Physics”, and it is characterized by a title and a brief description. Each topic is then divided into chapters.
- **Chapter** The chapters are characterized by a title, a description and an order that respects the traditional Italian study program. Each chapter is divided into lessons.
- **Lesson** A lesson has been created to emulate the contents of a traditional school lesson. Moreover, it provides support material and has been arranged facilitate users. It is characterized by a title, a description, an order, an abstract, a description, the corresponding school type (like “Liceo Scientifico”, “Liceo Classico”, . . .) and year of teaching (I, II, III, VI, V). Each lesson is composed by
  - **Posts** This is the smallest unit of text containing didactic material and it is characterized by a title, the content and an order.
  - **Levels** Consists in the exam material, that is a set of queries and possible answers grouped by difficulty. Each level is in fact characterized by a title, a description, a difficulty (1, 2 or 3), the total number of queries and the minimum amount of points needed to pass the level. Furthermore, a level is composed by a set of queries and possible answers.
    - **Query** A query represent an exercise and it is characterized by a text, an explanation of the solution and a type that reflects the form of required answer (true/false, open, single, multiple)
    - **Answer** It is characterized by a text, an order and a flag that establishes whether it is the correct answer to the associated query.

From the initial material we considered the lessons belonging to the “Scientific High School” (“Liceo Scientifico”) and discarded the topic Statistics (“Dati e Previsioni”) since it contains little content. We then analyzed the chapters and discarded redundant lessons, like review lessons, as well as lessons uncorrelated with respect to the chapter, like lessons about curiosities or fun facts.

Topics	Chapters	Lessons	Posts
Arithmetic and Algebra	21	119	388
Geometry	8	38	121
Relations and Functions	18	108	434
Physics	20	102	387
Total	67	367	1330

Table 1.1: Overview of analyzed lesson material.

In total, the perimeter of our study consists in 4 topics, 67 chapters and 367 lessons (see Table 1.1). This information will be used to model the content of the Redooc platform and build a Knowledge Structure (see Chapter 2). We then assigned an order to the content utilizing one of the attribute named present in each table used to determine the order of displaying the different elements into the pages visible to the students.

We also took into consideration a table containing to some extent the associations among the posts and lessons of the platform. Even though it is incomplete and not constantly updated, we used it to validate the Knowledge Structure that we will present in the next chapter.

1.2 User Information

By subscribing to the platform, users have the opportunity to take the exams and access certain didactic material. At the moment of inscription the user may provide their personal information, such as name, gender, birthday, and address, as well as information related to their education, such as their type (student, parent, etc.), their class if they are students, the institute name, the school type ("Scientific High School", "Classic High School", . . .), and the school year. These data are stored along with other information related to the platform, like the user's registration time and their total score gained on the platform. There are no constraints or verification at the time of registration, therefore the user information is typically incomplete and may be incorrect.

We analyzed the user information to identify the perimeter of our software:

- We considered the almost 340 thousand users subscribed to the platform and analyzed the distribution of the user type. As expected, we discovered that the majority of Redooc users are students (see Table

Type	Student	Parent	Professor	Minor13	Passionate	Alumnus	Null
Percentage	57%	20%	11%	7.3%	4.4%	0.016%	0.2%

Table 1.2: Distribution of user type.

Type	Male	Female	Other	Null	Total
Student	53.7	61.1	2.9	76	193.7
Parent	14.1	31.1	0.2	20.8	66.2
Professor	5.5	23.1	0.1	10.1	38.8
Minor13	8.8	8.2	0.1	7.5	24.6
Passionate	4.7	5.3	0.6	4.3	14.9
Alumnus	/	<0.1	/	<0.1	<0.1
Null	<0.1	<0.1	<0.1	0.7	0.7
Total	86.9	128.7	3.8	119.5	338.9

Table 1.3: Frequency of each gender per user type (in thousands).

1.2). We therefore decided to focus only on users labelled as student, the target of the platform and this project;

- We checked the user information to understand if there were other possible features related to their performance and behaviour. Unfortunately, no other field was taken into consideration due to the fact that there was plenty of missing data and some attributes could be unreliable (see Table 1.3 for the frequency of the genders per user type);

- We studied the number of daily subscriptions and noticed a seasonal behaviour, with subscriptions peaking in January and October, as well as a general increment during the last 2 years, due to the increase in popularity of the platform. Furthermore, we noticed a peak of subscriptions on March 14th of 2018 and 2019 (see Figure 1.1): this phenomenon derives from the special Pi Day challenge organized by Redooc, where subscriptions are free, and users are invited to complete as many exercises as possible. Particular attention will be paid to Pi Day, since users have a different behaviour compared to regular days.

1.3 Exam Material

At the end of each lesson a set of levels are proposed to test the user's comprehension of the concepts contained within the lesson. Each level is

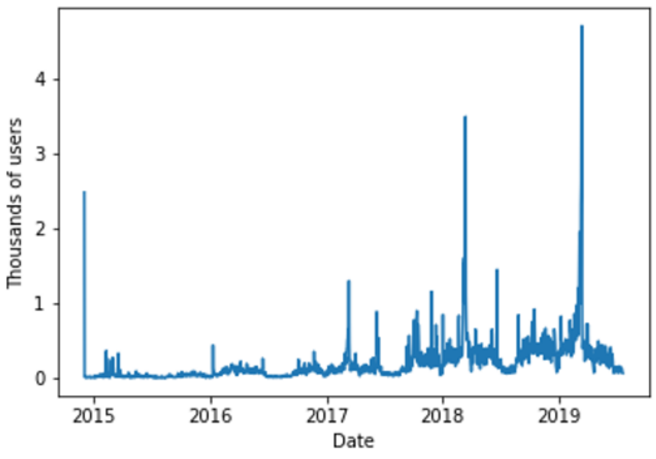


Figure 1.1: Daily frequency of subscriptions

associated to a certain difficulty, typically ranging from 1 to 3. By choosing a level, an exam is generated by randomly selecting a subset of the level's queries. During the exam, the user is presented with a series of queries and must face them one at a time, having up to two attempts to answer each query correctly. A query can be of different types: single answer (there is one correct answer among four possibilities), multiple answer (the correct answer is composed by exactly 2 of the 4 presented possibilities), True or False (the student has to decide between true and false) and open answer (the student has to write the response). After the user has completed their attempts, the correct answer is shown with a detailed explanation. Each query is associated to a score, equal to 2 times the difficulty level. If the student answers correctly at the first attempt, they will receive the full score; if they answer correctly at the second attempt, they will be rewarded with half the score; finally, they are awarded 0 points if they answer incorrectly both times. To pass an exam, it is necessary to obtain at least 60Each exam can be stopped at any moment, even between the first and the second attempt of the same query. If that happens, the status of the exam is saved, and the student can continue it another time.

The exams are recorded as "examlogs". Each examlog is characterized by an id, a creation time, a type (exercise, revision, classwork, homework, home study), the level identifier, the user identifier, the status (open, close, expire), the creation time, the last edit time, the number of answered questions, the number of correct answers, the score and a binary field that reports whether the exam was passed. Furthermore, each attempt at a query is archived as a "querylog". Each querylog record is characterized by an id, the creation time, the last edit time, the user identifier, the query identifier, the exam identifier, the correct field, the number of tries (0 if the user exits the exam before answering the query, 1 if it is the first attempt, 2 if it is the second attempt) and the score obtained on that query.

As anticipated, we focused on the students' activities related to the High School lessons of "Arithmetic and Algebra", "Geometry", "Relations and Functions" and "Physics". Furthermore, we took into consideration only terminated exams. We then carried out the following analyses on exam material:

- We studied the trend of the daily number of completed exams since the creation of the platform, totaling to around 140 thousand (see Figure 1.2). We immediately identified two peaks corresponding to the Pi Days of 2018 and 2019. Since a competition is held on these days, the objective of the students is not to effectively study the lessons but to accumulate as many points as possible. In fact, many students take exams of previous years or repeat the same exam several times. Moreover, we noticed that users that subscribe on Pi Day do not continue using the platform for a substantial amount of time. Since we are interested in the logics and mechanisms behind learning, we excluded these two days from the perimeter of our analyses.

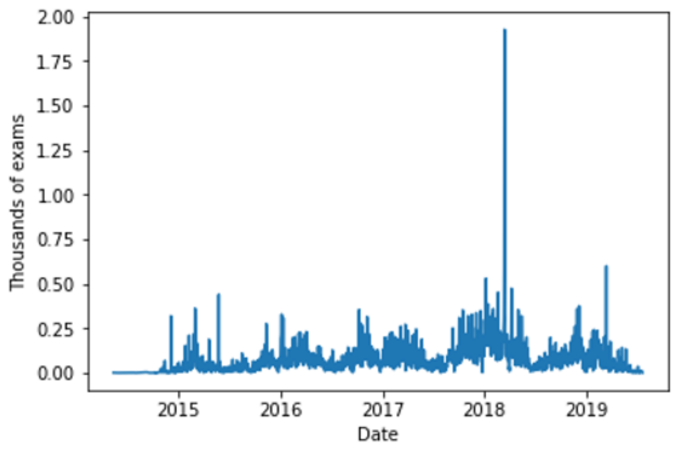


Figure 1.2: Daily frequency of completed exams

- After excluding these two days, the total number of exams is reduced to around 138 thousand. By re-plotting the daily number of completed exams, we noticed that, in correspondence to the increment of new subscriptions, there is an increment in the number of completed exams (see Figure 1.3). In particular the Redooc platform registered the highest employment during the scholastic year going from October 2017 to September 2018

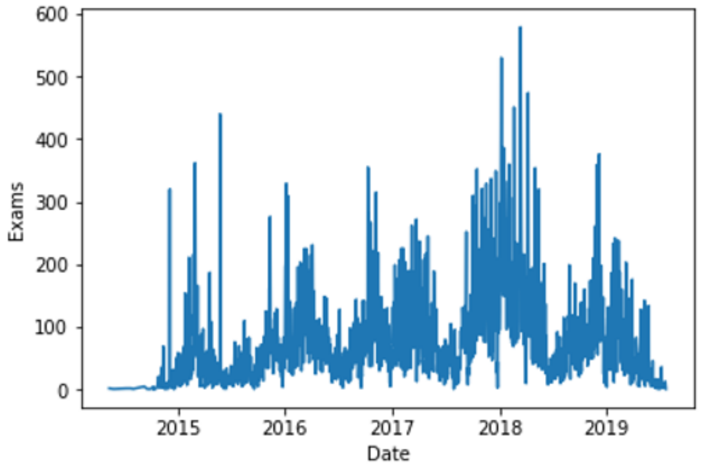


Figure 1.3: Daily frequency of completed exams (excluding 14/03/2018 and 14/03/2019)

By plotting the number of completed exams per month (Figure 1.4), we identified a seasonal behavior with intense usage of the platform during the fall and winter periods.

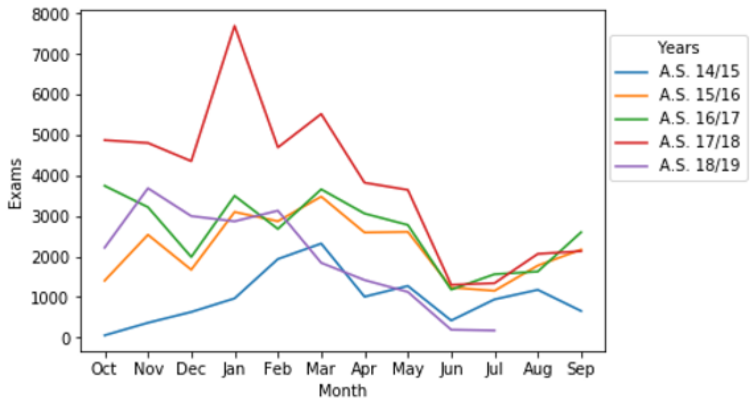


Figure 1.4: Monthly frequency of exams per scholastic year

- We analyzed the exams completed per user and discovered that many students utilize the platform only for a reduced number of exams (see Figure 1.5) In fact, even if there are students with more than 300 completed exams, the average number of completed exams per student is around 8.

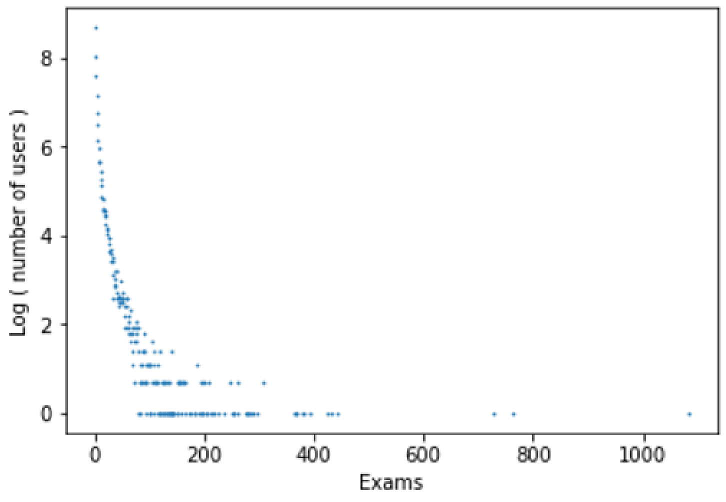


Figure 1.5: Logarithm of the frequency of students per number of completed exams

We then shifted the focus of our study from the exams to the queries. Since the creation of the platform in 2014, almost 4 million querylogs have been generated with users answering correctly 60% of the time on the first try and only 48% on the second. We gradually filtered the querylogs according to the following analyses:

- We did not consider the querylogs related to queries that have already been attempted over four times by the same student. By continually repeating an exam and the queries, a student could easily memorize the solutions without effectively studying and learning the lesson content.

- We considered only the first try since the second attempt is far less indicative of the real knowledge of the student. Furthermore, even without knowing anything about the topic, the odds of guessing the solution on the second attempt are higher.

- We considered only students which have completed at least 48 queries. This allows us to focus on students which are active on the platform and therefore study their didactic progress. The Adaptive e-Learning System in fact aims to guarantee the best educational support to those students that use the platform frequently. In order to justify this choice, it is useful to consider Figure 6:

- We first studied the users' activity. The blue line shows how the percentage of Redooc users varies in terms of the minimum amount of completed queries. In other words, given a number of querylogs, we calculated how many students completed at least that number of querylogs. There is an exponential decrease that confirms the trend shown in Figure 1.6;
- We then evaluated the amount of data (in terms of querylogs) corresponding to a certain level of user activity. The orange line shows how the percentage of querylogs carried out by the users varies in terms of the users' activity on the platform. In other words, given a number of querylogs, we calculated the total amount of querylogs carried out by users that have completed at least that number of querylogs. This time there is a linear decrease.

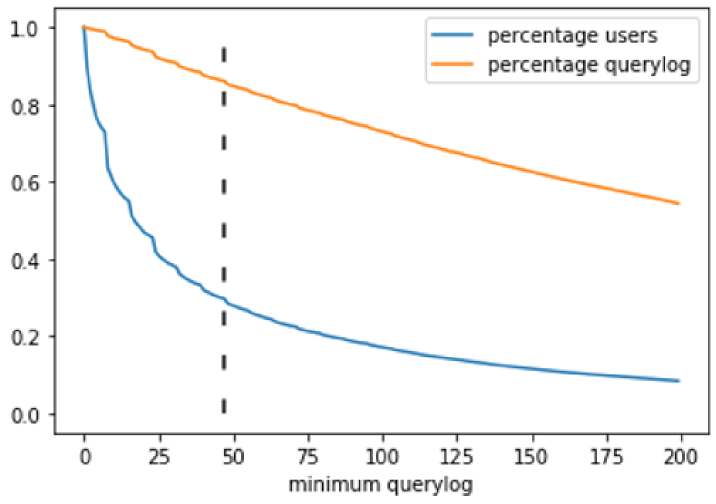


Figure 1.6: User activity and the corresponding querylog data

These two trends allow us to take into consideration only the subset of students that frequently use the platform without losing too much information coming from the queries. In fact, by setting the threshold to 48 queries (dashed line in Figure 6) we are still taking into account more than the 90

By applying these filters, we obtained the exam data that will we used to model the students' knowledge (described in Chapter 3). The perimeter of our study therefore consists in almost 2 million querylogs, carried out by

Year	I	I, II	II	II, II	III	IV	V
Percentage	58.0%	0.7%	17.0%	7.0%	10.0%	3.6%	3.7%

Table 1.4: Distribution of querylogs per year.

Topic	Percentage
Arithmetic and Algebra	69%
Geometry	12%
Relations and Functions	17%
Physics	2%

Table 1.5: Distribution of the querylogs per topic.

almost 9 thousand students and covering 331 lessons of the 367 identified in Chapter 1.1. We repeated the previous analyses on this data and obtained the following results:

- Users answer correctly 58% of the time;
- We analyzed the distribution of querylogs among the different school years (see Table 1.4) and noticed that the majority of students carry out queries belonging to the "biennio", that is the first two years of high school. In fact, more than 50% of querylogs belong to the first year, while almost 75% belong to the first 2 years.
- We analyzed the distribution of querylogs among the different topics (see Table 1.5) and discovered that the majority belong to "Arithmetic and Algebra". This is coherent to what we have found with the analyses of the distribution of queries among the different school years, since most of the lessons of the first and second year regard "Arithmetic and Algebra".
- We analyzed students' permanence on the platform by analyzing the churn rate (see Figure 1.7). More specifically, we studied the percentage of students in function of the minimum number of days spent on the platform, calculated as the time between the first and last querylog. Even though we are considering students that have been active on the platform (at least 48 completed queries), the steep increase of the curve highlights that many students stop using the platform after a short period of time. In particular almost 20% of students stop using

the platform in 2 days from the first usage and almost 90% are active less than a year.

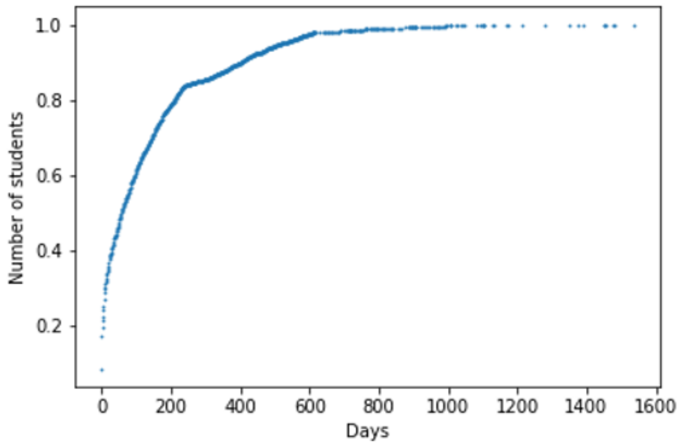


Figure 1.7: Students' churn rate

1.4 Wikipedia Data

In general, Machine Learning models require a large amount of data. Text mining algorithms and Natural Language Processing are particularly sensitive to the amount of data since they must search for patterns in noisy, unstructured data. We therefore decided to enrich our initial corpus of lessons by downloading Wikipedia pages with content related to the analyzed topics of the platform. In order to do this, we used 2 approaches:

- We identified the Wikipedia pages directly related to the content of the platform. This was done using the Wikipedia API for Python (https://wikipedia.readthedocs.io/en/latest/quickstart.html), giving as input the 331 lessons titles, the 67 chapters titles and the 4 topics titles
- We identified the Wikipedia pages containing keywords related to the content of the platform. This was done using the website https://petscan.wmflabs.org/: given a keyword, it provides all Wikipedia pages which are linked to that keyword and satisfy certain filters. We focused on the pages that are directly linked to the Italian Wikipedia pages of "algebra" ("algebra"), "aritmetica" ("arithmetic"), "combinatoria" ("combinatorics"), "fisica" ("physics"), "geometria" ("geometry"), "matematica" ("mathematics"), and "statistica" ("statistics").

> Once we obtained the related titles, we downloaded the Wikipedia pages using the API as in the previous approach

In total we downloaded almost 8 thousand pages, which we used to enrich the corpus of texts and build a more stable Knowledge Structure (see Chapter 2).

Chapter 2

Adaptive e-Learning System

In recent years, schools and the education system in general have seen the spread of new didactic tools as well as the creation of innovative ways to deliver educational content worldwide. Technology is having a profound impact on learning, by providing access to more information, improving the quality of learning and introducing new untraditional teaching methods. The fusion between education and technology required us to re-elaborate the foundations and the strategies of the whole education sector. The amount, the quality and the availability of the data have increased over the years, giving the possibility to receive and transmit information in new ways.

In particular, distance learning allows students to sit in a remote location while interacting inside virtual classroom environments. Starting from the first and simple online courses, we soon arrived at what today represents the frontier of distance learning, that is MOOC or Massive Open Online Courses [**1**]. They are interactive online course aimed at unlimited participation and open access via the web. In addition to traditional course materials (lectures, videos, exercises, etc.), many MOOCs provide interactive courses as well as user forums and social media discussions to support community interactions among students and professors and to allow fast feedbacks to questions and assignments from teachers, other users or even automated systems [**25**].

The creation of such systems, starting from the educational content up to the development of the platform, is becoming progressively more important. In particular, these systems are evolving towards the creation of personalized environments, where the automation of the adaptive part is increasingly required. Hence the birth of what can be called Adaptive e-Learning Systems (ALS), in which the central figure is no longer that of the teacher, but rather that of the student. In fact, these systems aim to provide each student with the best individualized educational experience possible. The personalization of an ALS ranges from the web structure of the platform to the actual contents and aims to guide the user along the optimal individual learning path by proposing the best material according to the users' characteristics. These characteristics may include the users' preferences, goals, behaviors, actual knowledge, learning speed and needs [**2**].

In the following section we will present the basis of the development of an Adaptive e-Learning System, the general architecture representing such system and the steps needed to implement it.

2.1 Adaptation principles

What are the final goals of an ALS? Which aspects we have to take into consideration when we implement it? Are there general rules, guidelines or restriction we have to follow? How is it possible to test the goodness of the system? What does "best learning experience" mean and how we can define it?

These are some of the main questions involved in just the definition of an ALS. The problem is so generic and complicated that a clear and usable protocol has not been defined to date in any context. Indeed, there are many variables involved: different students, different subjects, different means of teaching, ... Not to mention the fact that all these variables continuously vary over time. The fact that a student continuously learns (or forgets) new notions over time extremely complicates the task. We don't have access to informations regarding the life of the students outside the activities they have completed on the platform: have they studied the new arguments? have they forgotten a particular concept? And, even knowing these data, there is still the problem of extrapolating and analyzing meaningful information.

However, when designing an ALS, it is important in general to take into consideration the following aspects:

- what it is appropriate to adapt
- when it is appropriate to adapt
- how it is appropriate to adapt

2.1.1 What to adapt

This question reflects the necessity of simplify and schematize the general problem in order to be able to construct a model applicable in any situation. This is the core aspect of the entire system. The response of this question is strictly correlated to the goal of the system and determines the building blocks of the entire architecture.

First of all, it is necessary to distinguish between the adaptation possibilities with respect to the users and the adaptation possibilities with respect to the platform.

Users

More in detail, in the first case we refer to which aspects of the users has to be taken into account when evaluating the adaptation. Moreover, do these aspects differ among the users?

The individualization of the most meaningful users' features that can be used to model them is the first step in the creation of the users' profiles that will help us in the adaptation process. It is possible to identify each user with the following personal features:

- knowledge - This aspect is related to the actual level of knowledge of each student with regards the topics of the educational material presented in the platform. It is the one of the most important and most considered aspect in the development of an ALS. Many adaptive systems are based on this feature and consider it as a source of adaptation. Even if this is the most logical feature to consider in an educational framework, it is also the most difficult to evaluate. In fact, the system should be able to recognize the initial knowledge of each user, its changes over time and be able to update the model accordingly to it;

- goals - This aspect is related to the reasons why the user is using the platform. Even this motivations could change over time, so it is important to build a system that is able both to represent and understand them in any situation. Like during the real courses, the goals could be assigned not by the student himself, but by other peoples. In our case it is possible to identify 4 main people related to the various assignments: the teacher, course author, the system developer and the student himself;

- experience - This aspect is related to the degree of use of the platform. It reflect both the level of experience in using the platform and the activities performed on it. Its evaluation directly derives from the recording of all the actions of the users within the platform.

- background - This aspect is related to the experiences of the user outside the platform. Knowing the starting point of each users any time they use the platform is extremely important in the adaptation process. Even systems able to adapt extremely quickly to the users' changes will initially provide incorrect results if the background evaluation is inaccurate. This fact influences significantly systems where the average time of usage of the platform during a single session is very low (like in our case, see section 1.3).

- preferences - This aspect is related to the personal preferences of each user. It is common to find students with different inclinations or predilections for certain topics. Presenting an argument in a particular way or setting a learning path more comfortable to the students could affect significantly the degree and the speed of learning. Not to mention the fact that following the users' preferences could increase the usage of the platform, that, from an economic point of view, could be one of the main targets for the platform owner.

Platform

In the simplest scenario, an educational platform could be structured as a set of educational elements connected by links. Then, each page visible by the students corresponds to local educational informations and several links to other pages.

In this case, the adaptation regards two main aspects:

- presentation - This aspect regards specifically which educational contents and how they are presented to the students according to their features. In addition to the contents that are presented, the way they are presented represent an important aspect in a learning environment;

- navigation - This aspect regards the support that the students receive during learning. In fact, with an appropriate navigation support it is possible guide and help the students in retrieving information, searching for specifics contents, following a specific learning path.

2.1.2 When to adapt

The adaptation times may regards components of the ALS both visible and not visible to the users. The first care mainly regards the platform content and links that are shown at the creation of each different page, while the second regards all the underlying features, as the users characteristics, needed to perform the adaptation.

The timing in which an adaptation is performed has to be defined during the creation of the ALS and it depends on several factor:

- users' interactions - Each time a student interact with the system it could be necessary an adaptation. The simplest case regards the change of the page through the usage of a link. In this circumstance, not only new contents have to be displayed, but also the users features could require an adaptation. In fact, for example, if the new page has included less advanced topics, it could mean that the student has gaps in those particular topics.

  For the system even the absence of interaction for a long period of time could represent a source of information about the student. For example, a long period of time spent on the same lesson or query could represent the difficulty of the student in a particular topic and the need of assistance through support material like concepts maps or hints.

- time-space limitations - Each recorded data or system's feature requires space and each recording or adaptation requires time to be performed. Even with infinite space availability, in a thousands of remote users systems like ours time represents a serious problem. In many cases the adaptation is somehow reduced or interrupted for the sake of the users' experience on the platform. For this reason, simpler and faster models are preferred. These aspects are extremely relevant in such and

- experts theories - The recent increase of the interest for the educational sector has attracted not only classical professional figures like teachers or psychologists, but also statisticians and data scientist. In fact, the increment of the amount and the availability of data in this sector has allowed the conduction of more and more studies until the birth of a new emerging discipline, the Educational Data Mining.

  Then, it is possible to follow the theories of the experts of different fields in order to provide the best educational experience.

- users' settings - Finally, it is even possible to some extent to leave the choice of when and what to adapt to the users.

2.1.3 How to adapt

In this section we will present which are the main ways an adaptation could be performed and presented to the users. The discussion about the underlying motivations concerning the possible models implemented will be presented later.

The following are most common sources of adaptation [**3**]:

- content adaptation - Even if the educational material could not be changed, it is possible to present or not present it in different ways or in different orders, like me mentioned in section 2.1.1;
- adaptive annotations - They consist in visual cues that the users can attach to links in the current page they consider important. These annotations help the students' orientation in the page and speed up the learning;
- recommendation system - It is possible to develop a recommendation system that takes into account the users features and builds links to pages that are not directly reached by the current page and that are relevant for the user;
- feedbacks - It is possible to provide intelligent feedback to the users with the aim of encouraging, motivating, making them use more the platform, keeping them updated on their performances and maximizing the learning progress.

2.2 ALS base architecture

The project related to this thesis consists in the development of an ALS starting from a pre-existing platform already provided with the educational material and particular content presentation and navigation support rules. For this reason, we will focus only on the aspects that are relevant for the development of an ALS in a similar context.

Then, once individualized the main available features that have to be considered in the development of the ALS, it is necessary to structure them in order to create a model that reflects the scopes and the desired usage of the system.

In our case, one of the main scopes is the understanding of the users' lack of knowledge in the various topics and the evaluation of the optimal learning path able of guarantee a solid preparation in each argument encountered during the school lessons. Hence, the ALS has to take into consideration the actual knowledge of each student.

The creation of a system with this characteristic involves the study of the following three aspects about the knowledge modelling:

- Domain Structure – the domain structure is a set of small intercorrelated knowledge elements or concepts specific to a particular domain. Each concept represents an independent elementary fragment of knowledge for the given domain, therefore denoting the smallest unit of knowledge that can be isolated. Once the set of concepts or domain knowledge has been identified, the concepts can be linked by relationships of any kind. An example of relationship is be the "prerequisite" relation, where a first concept is the prerequisite of another if and only if it is necessary to understand the first to understand the latter;

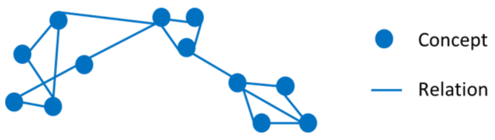


Figure 2.1: Domain Structure

- Educational Structure – the educational material is represented by the elements and materials belonging to the system that have an educational impact on the users (for example notes, readings, videos, images, schemas, exercises, challenges, etc.). Since the goal of these elements is to transmit knowledge, they are based on the concepts of the related domain. Therefore, there exists a structure, which we will call educational structure, underlying the educational material that implicitly inherits the one belonging to the domain structure. In fact, each educational element can be expressed as a combination of concepts and each relation among them is a relationship obtained from the related domain structure;

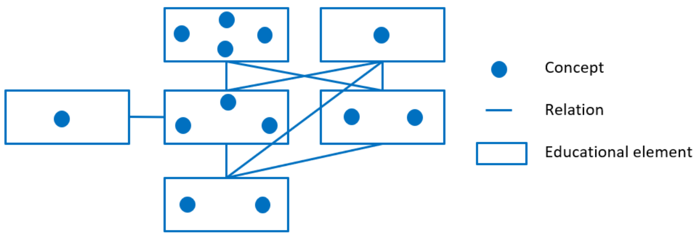


Figure 2.2: Educational Structure

- Platform Structure – given the system's educational material, platform development is the way the material is presented and connected inside the system's platform through implemented functionalities. Platform development can be divided into functionalities related to the presentation of the material, involving the way the educational elements are displayed (such as order, font, color, and size), and the navigation of the material, that involves any sort of relation among the educational elements (such as links, menu bars, research bars, pop-ups, etc.). By personalizing these features on the user and having them change over time, the adaptive element is introduced. Then, based on the platform development, it is possible to design the platform structure: the way each educational element is presented can be interpreted as a platform element, while the implemented navigation possibilities correspond to the relationships among the platform elements. The presence of adaptive functionalities leads to an adaptive platform structure that provides each student with a personalized educational experience.

The definition of these three aspects corresponds to what we will call the Knowledge Structure of a specific Adaptive e-Learning System. Then, in order to build the best possible learning system, it is necessary to identify the underlying domain knowledge and domain structure, to select the educational material and create the educational structure, and finally to develop the platform structure with adaptive functionalities.

The creation of this structure involves different aspects regarding the relationship with the Domain Structure:

- Cardinality

  It regards the the cardinality of the possible relations between the concepts space and the elements of the educational platform. Essentially it is divided into 2 main approaches:

  - single concept indexing - Each fragment of educational material is related to one and only one domain model concept. Is is simpler and more intuitive.
  - multi-concept indexing - Each fragment can be related to many concepts. It is more powerful, but it makes the system more complex. In many cases using a multi concept indexing is imposed by the nature of the domain. For example, in programming and mathematics most of the examples and problems involve several constructs and operators.

- Expressive power

  It represent the amount of information associated to each link between a concept and a page. Just the presence of a link between a concept and an element of the educational platform is an important information, but in many cases it is possible to associate more features to every link by using roles (like prerequisite, summary, explanation, introduction of) and/or weights (like the percentage of knowledge about a concept presented on this page) in order to allow the implementation of more advanced adaptation techniques.

- Granularity

  It is the precision of labelling. It is possible to index with the same concepts a whole page, a fragment of a page or a cluster of connected pages.

- Navigation

  It concerns the links between a concept and a page. They exist only on a conceptual level and are used only by internal adaptation mechanisms of the system. Moreover, they can be used to define the various navigation learning paths.

Then, the different choices made by the developers during the creation of the Knowledge Structure largely defines the typology and the functionalities of the ALS.

In fact, the Knowledge Structure directly define to large extent the student' model. In particular, it provides a framework for representing of the user's knowledge through the usage of the Domain Structure.

The most important users' knowledge models are:

- overlay model - For each concept the model stores some data that represent an estimation of the user knowledge level for that particular concept. In the simplest form it is a binary value: known or not known.

- weighted overlay model - It distinguishes several levels of user's knowledge of a concept using a qualitative value (for example, good-average-poor), an integer quantitative value (for example, from 0 to 100), or a probability that the user knows the concept.

- historic model - It keeps some information about user visits to individual pages such as the number of visits or time spent on a page. This model could be implemented alongside the others and could represent a different source of features for the ALS.

Also the others users features (goals, experience, background, preferences) could be expressed through the usage of the Domain Structure: a goal could be represented as a set of different concepts to learn, a preference could be represented as a set of different concepts more comfortable to the user, the experience could be estimated by the activities performed by the student on the platform and the background could be evaluated using periodical general tests that help the system to understand the level of knowledge of the user even without knowing his/her experiences outside the platform.

There are many other possibilities to model these students' features (for examples many MOOCs use the users ratings to understand their preferences), but we have not considered them since there are no available data to allow their implementation, like we have seen in Chapter 1.

For this reason, we elaborated the best course of action in the development of the ALS with the available data and accordingly to the scope of the project, trying to extract as many information as possible from the users' activities.

2.3 Implementation Steps

The collaboration with Redooc led to the creation of an Adaptive e-Learning System by introducing an adaptive functionality Cassandra to the platform (Figure 2.3).

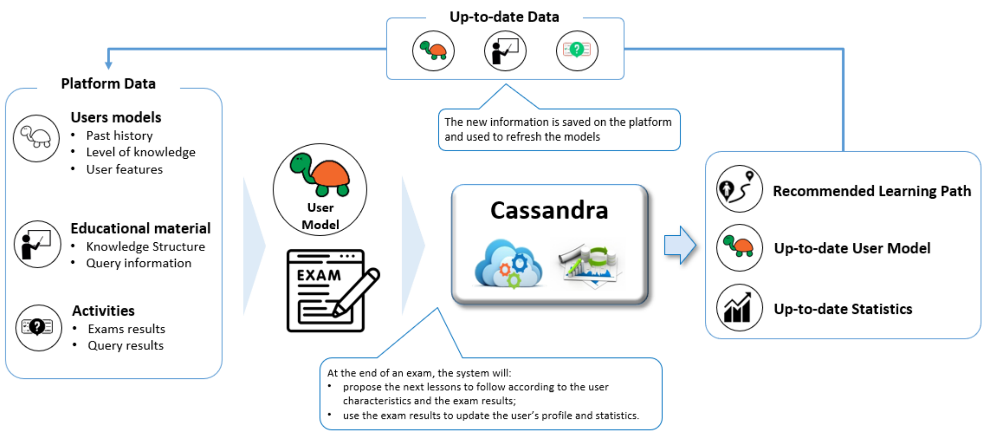

Up-to-date Data
Platform Data
Users models
Past history
Level of knowledge
User features
Educational material
Knowledge Structure
Query information
Activities
Exams results
Query results
User Model
EXAM
The new information is saved on the platform and used to refresh the models
Cassandra
Recommended Learning Path
Up-to-date User Model
Up-to-date Statistics
At the end of an exam, the system will:
propose the next lessons to follow according to the user characteristics and the exam results;
use the exam results to update the user's profile and statistics.

Figure 2.3: Adaptive e-Learning System with the adaptive Cassandra process

Cassandra consists in a recommender system for high school students navigating the Redooc platform, that indicates at the end of an exam the next lesson the student should study in order to maximize their learning. The system is based on the student's exam results, where a model is able to evaluate the student's knowledge of fundamental mathematical concepts.

Furthermore, the recommendations must respect the underlying educational structure. The concepts and educational structure were developed starting from the platform material and have been summarized in a Knowledge Structure, while the user model was developed utilizing Machine Learning algorithms. Finally, since the recommendations are proposed in the form of links to the suggested material, the system impacts only the navigation aspect of the platform.

In the following chapters the methodologies and steps utilized for the creation of Cassandra will be analyzed and explained more in detail. In particular, in our case, the whole process can be divided into three main stages (see Figure 2.4):

- Construction of a Knowledge Structure – consists in the creation of the domain structure (that is an intercorrelated collection of concepts specific to a particular domain) and a new educational structure (that is an organization of the educational material existing on the Redooc platform) related to high school mathematics;
- User modelling – consists in the modelling of each student's knowledge and skills, utilizing information about their exam results;
- Creation of a Recommender system – consists in developing an algorithm capable of guiding students along their personalized learning path, by suggesting the optimal lesson to study based on their skills.

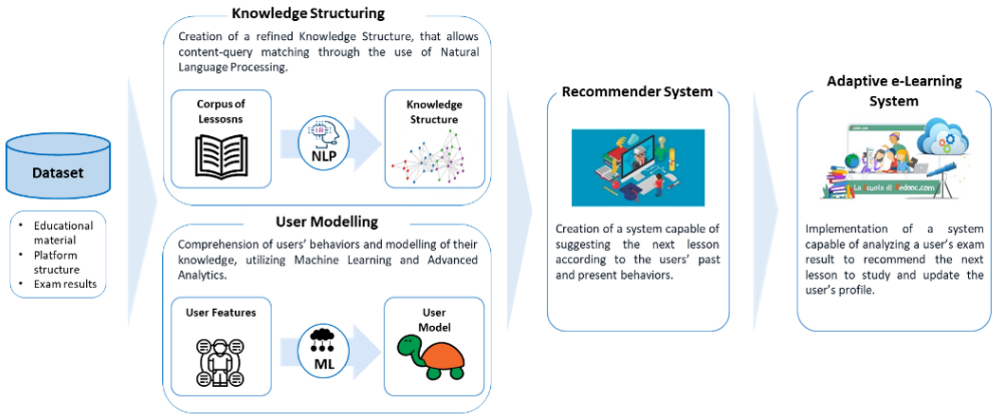

Dataset
Educational material
Platform structure
Exam results
Knowledge Structuring
Creation of a refined Knowledge Structure, that allows content-query matching through the use of Natural Language Processing.
Corpus of Lessosns
NLP
Knowledge Structure
User Modelling
Comprehension of users' behaviors and modelling of their knowledge, utilizing Machine Learning and Advanced Analytics.
User Features
ML
User Model
Recommender System
Creation of a system capable of suggesting the next lesson according to the users' past and present behaviors.
Adaptive e-Learning System
La Scuola di Redooc.com
Implementation of a system capable of analyzing a user's exam result to recommend the next lesson to study and update the user's profile.

Figure 2.4: Adaptive e-Learning System creation process

Chapter 3

Knowledge Structure

In order to build an adaptive learning system based on the users' knowledge, it is necessary to identify the underlying domain structure specific to the particular domain and create an educational structure starting from a selection of educational material. The domain structure is a set of intercorrelated concepts specific to a particular domain where each concept represents an independent elementary fragment of knowledge for the given domain. Once the set of concepts or domain knowledge has been identified, the concepts can be linked by relationships of any kind (for example the "prerequisite" relation). An educational structure instead consists in the organization of educational material that implicitly inherits the relations belonging to the domain structure. In fact, each educational material can be expressed as a combination of concepts and each relation among them is a relationship obtained from the related domain structure.

In our case, we were provided with the educational material belonging to the Redooc platform and an incomplete educational structure. Since Cassandra's recommendations must be based on the educational structure in order to tutor the students in the best way, we developed a procedure to create the complete educational structure. The procedure required identifying the domain knowledge and prerequisite relations starting from the platform's Lesson material and the Wikipedia data (see Chapter 1.1 and 1.4). By combining this information with the platform's initial educational structure, we created a new educational structure which we defined as Knowledge Structure. Furthermore, we extracted additional information to support the construction of the recommender system, such as the level of knowledge provided by each educational element and the level of knowledge required to answer each query correctly.

To create the Knowledge Structure, we must leverage on Natural Language Processing (NLP) techniques and algorithms. NLP is a branch of artificial intelligence that deals with analyzing, understanding and generating the human language. Standard NLP techniques require a text corpus, that is a set of documents containing large amounts of textual, unstructured data. For the construction of Cassandra, we constructed a corpus based on the following documents:

- Redooc posts, utilizing the title, description, and abstract;
- Redooc lessons, utilizing the title, description, and abstract;
- Redooc queries, utilizing the query text;
- Redooc answers, utilizing the text of each possible answer and the explanation of the correct answer;
- Wikipedia pages, utilizing the title and content.

As mentioned in Chapter 1.1, we considered the lessons belonging to the "Scientific High School" ("Liceo Scientifico") and with topics "Arithmetic and Algebra", "Geometry", "Relations and Functions", or "Physics". We also discarded redundant lessons, like review lessons, as well as lessons uncorrelated with respect to the chapter, like lessons about curiosities or fun facts. In total, the perimeter of our study consists in 4 topics, 67 chapters and 367 lessons.

After identifying the documents that build the corpus, we built the Knowledge Structure and extracted additional information by applying the following steps (Figure 3.1):

- Preprocessing – consists in the transformation of the corpus of texts, that are our initial educational elements, into mathematical objects that can be easily analyzed and used as input for NLP algorithms;
- Structuring – consists in the definition of the base concepts and the identification of the relations among the educational elements, therefore building the new educational structure;
- Knowledge modelling – consists in the creation of a model capable of determining the level of knowledge provided by each post and lesson;
- Query-Lesson association – consists in determining of the level of knowledge required to answer each query correctly, obtained by evaluating the level of knowledge provided by the lessons that are most similar to the query.

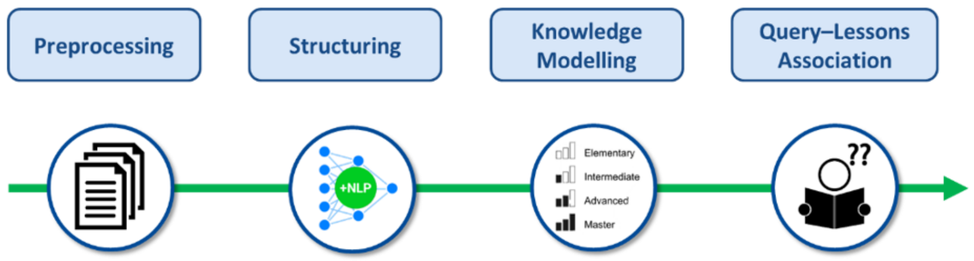


Figure 3.1: Knowledge Structure creation process

3.1 Preprocessing

Texts represents large unstructured data, difficult to handle mathematically. For this reasons, various preprocessing procedures must be carried out to obtain structured information elements that can be more easily and effectively modeled. These procedures could be summarized into the following steps:

- Text preparation – consists in the set of procedures that transform the initial corpus of texts into a form that is predictable and analyzable. Some examples of text preparation methods are lowercasing, stemming, and tokenization;
- Text structuring – consists in representing the output of the previous step in a mathematical form capable of modelling and summarizing the main features of the corpus.

3.1.1 Text Preparation

Text preparation is typically composed by the following procedures:

- Text formatting – the set of processes that transform the texts into a unique coherent format. In our case, many of the texts contained expressions deriving from a HTML format or formulas in the LaTex format. Since these expressions are not useful, or may even have a negative impact, we decided to convert them in a human readable form or directly delete them. Other types of text formatting used are the conversion of all characters into ASCII code, the conversion of written numbers to the numeric form, and the lowercasing of all texts;
- Tokenization – process of splitting the text into words, phrases, symbols, or other meaningful elements called tokens, eventually replacing certain input with ”tokens” which represent their meaning. “New

York" for example can be treated as a single word, while "I'm" can be separated into the two words "I" and "am". The purpose of tokenization is to simplify content prior to the next step of processing;

- Noise removal – consists in filtering out all the tokens that are unnecessary, like the stop words, or that worsen the text structuring. In fact, in order to understand a text not all words or punctuation are necessary. For example, if we do not consider the commas inside a sentence its sense may be equally understandable. For this reason, we established a list of elements that have been deleted from the text corpus;

- Stemming – consists in the process of reducing a word to its word stem, base or root form. In fact, similar considerations to those made in the previous steps could be made about the internal parts of the words. Consider for example the use of certain verbs, where the final part is unnecessary in order to understand the meaning of the action. Expressions like "look", "looks", "looked" express the same action but they are represented by different words. A possibility is to merge all of these expression into a unique one. This procedure simplifies the texts and reduces the proliferation of many different words that share the same meaning and have the same lexical root. Stemming is very important when we do not have vast amounts of documents to analyze or when utilizing a complex language, like Italian

- Lemmatization - an alternative to the stemming procedure is the lemmatization procedure. As the stemming procedures, its scope is to extract the base lemma of each word. However, instead of representing each word with a truncated form like in the stemming process, it expresses words with the same lexical base through the usage of the representative lemmas.

- Part of speech tagging (POS) - consists in the evaluating the semantic meaning of the words into the phrases. The words are tagged with the essential semantic meaning like noun, verb, adverb, adjective and so on.

A qualitative analysis of the results of the application of these procedures conducted us to the usage of only the first 4. In particular, even if sometimes inaccurate, the best stemming results were obtained by the Snowball Stemmer of the nltk library. The fact that the texts are in Italian, not a common language in the programming environment, the fact that it is a complex language and the fact that the texts contains a very peculiar language, the scientific one, influenced the outcome of all the procedures. There are not many online packages with the Italian language incorporated and they are imprecise.

The lemmatization process labelled words with different lemmas through the usage of the same root and words with the same lemmas through the usage of the different roots. Just the Italian "superlativo" form was sufficient to identify in a different manner words with the same basic lemma. Not to mention the different conjugation of the verbs.

Instead, the POS process was intended to be used as an identifier of the verbs inside the phrases in order to be able to select the useless ones and improve the future models' performances. In fact, sometimes the verbs are not needed to understand the meaning of a phrase in a mathematical context. Even in this case there was an inversion of the outcomes: maybe the most representative incorrect case is the labelling of "retta" ("line") as a verb, probably due to the fact that it express also the past participle of the verb "reggere" ("hold").

3.1.2 Text Structuring

The final preprocessing step consists in the creation of a mathematical structure to model the data and make it more mathematically treatable. This can be done by introducing a vector space model, that is an algebraic model for representing text documents as a vector of identifiers (see Figure 3.2). Each document is represented as coordinates corresponding to a sequence of terms that often coincide with the words in the document's corpus or, more in general, with the keywords of the vocabulary.

Based on how the coordinates are calculated, we obtain a specific vector space model:

- One-hot encoding is the simplest way of representing the words. It is based on a vocabulary, that corresponds to the set of all the words in the corpus of texts. Then this model represents the words through the usage of binary vectors of size equal to the cardinality of the vocabulary. After a certain fixed ordering of the vocabulary, the i-th word is represented by the vector $[0, 0, ..., 0, 1, 0, ..., 0, 0]$, that is a vector of only zeros, except for a 1 in the i-th position. Even though it is a simple model, it has many drawbacks: it does not take into consideration the meaning and the context of the words in the phrases and it requires an high dimensional space with sparse vectors, making it inappropriate for many machine learning algorithms;

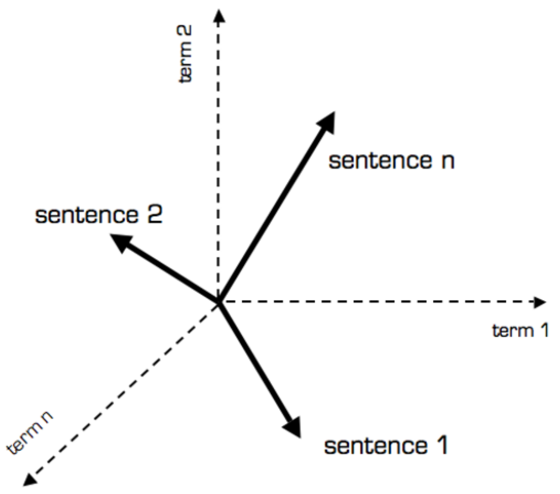


Figure 3.2: Vector space model

- The Bag-of-Words model (BoW) represents each document (or sentence) as a multiset of words. In particular, it represents a text as a vector where each term is the count of each word in the corpus.

  Let C a corpus of D documents and let N the size of the ordered vocabulary deriving from C, that corresponds to the number of unique tokens in C. Then the i-th document could be represented as a vector of size N in the following way:

  $$[f_{i,1}, \ldots, f_{i,N}]$$

  , where each term $f_{i,n}$ represents the frequency of the n-th word of the vocabulary in the i-th document.

  This model is mainly used for feature generation: transforming each text into a vector, we can calculate various measures to characterize the text. The most common type of features is term frequency, i.e. the number of times a term appears in the text. However, even this model has some drawbacks: it still requires a pretty high dimensional space and it does not consider the order of the words in the phrases and is completely based on the word frequency, that sometimes is not correlated to the importance of the word. The following model tries to take into consideration also the latter;

- The Term Frequency - Inverse Document Frequency model (TF-IDF) calculates the coordinates based on how important a word is to the document. It not only considers the frequency of words into documents, but also takes into account the occurrence of a term in the whole corpus in order to give more importance to words that are infrequent and penalize ones that are common.

With the notation introduced before, each element of the document vector is represented by:

$$\frac{f_{i,n}}{f_{i,*}} \log \frac{D}{D_n}$$

where $f_{i,*}$ represents the number of words in the i-th document and D_i represents the number of documents that contains at least once the i-th word of the vocabulary.

The first fraction correspond to the time frequency (TF) term, or the relative number of times that a word appears in the document. The second term correspond to the inverse document frequency (IDF) term, or the inverse frequency of a word among all the documents.

Then, by applying these models and vectorizing each word and document of the corpus, we were able to analyze the words' frequencies and importance both in general and in relation to a single document. At the end of the preprocessing phase we were thus able to discover:

- Terms that are frequent but seem useless in defining the meaning of the documents. Some examples are "risolvere" ("to resolve"), "cioè" ("i.e."), and "allora" ("then"). These terms increased the complexity of the models without adding value, so we decide to remove them from the corpus;
- Terms that are rare and seem useless in defining the sense of the documents. These terms are often part of the text of the queries where example scenarios are presented. We therefore decided to ignore them;
- Terms that are rare but seem extremely useful in defining the sense of the documents and their content. An example is "cotangente" ("cotangent") that immediately connects the text to trigonometry;
- Important terms that were misspelled, which we corrected;
- Important bigrams, that is words that often appear coupled and that must be considered as a unique term with its own meaning. The most important example is "radice quadrata" ("square root");
- The most important words in general, segmented per topic (see Figure 3.3).

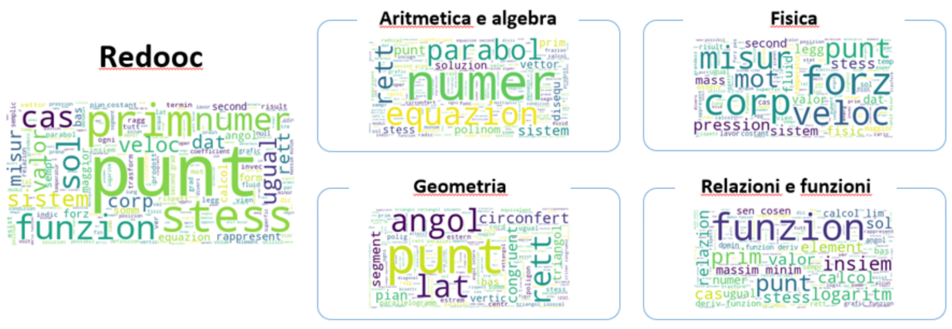


Figure 3.3: Word clouds representing the most important words

3.2 Structuring

The adaptive learning system must be able to predict the optimal learning path for each student. Since the concepts are primarily introduced inside the lessons, we decided to base the recommender system on the lessons: a recommendation therefore corresponds to a lesson that the student has the liberty of following.

When selecting a lesson to propose, the recommender system must have an underlying structure with specific ordering, rules and relations. This structure corresponds to the previously described educational structure with prerequisite relations: lesson A is a prerequisite of a lesson B if and only if, without considering other information, it is necessary to have learned lesson A in order to understand lesson B. We modelled the structure and relations with the usage of a simple directed graph where the vertices represent the lessons and the edges represent prerequisites. In other words, given the vertices X and Y, the edge (X, Y) directed from X to Y exists if and only if the lesson associated to vertex X is a prerequisite of the lesson associated to vertex Y. In order to create this structure, we followed two main approaches:

- Topic modelling – consists in the extraction of the main concepts from the corpus of texts and the expression of each document as a combination of these topics to form the final educational structure;

- Word embedding – utilizing specific models, we assigned to each word a vector which stores a set of hidden features able to uniquely define it. Then, we combined these vectors in order to assign a unique vector to each document. Finally, as in the previous approach, we were able to establish the links and the directions by evaluating the similarity among these vectors and following the order of the lessons.

3.2.1 Topic modelling

In NLP a concept model is a type of statistical model for discovering the abstract topics that occur in a collection of documents. We used these models in order to define the base concepts underlying the corpus of texts and then, in following step, utilize them to structure the educational material. This approach corresponds to the creation of an educational structure starting from the domain structure.

The most popular Topic Modelling Algorithms include:

- Latent Semantic Analysis or Latent Semantic Indexing (LSA or LSI)
- Latent Dirichlet Allocation (LDA)

These are unsupervised text analytics algorithm used to discover the underlying concepts of each document in a corpus of texts. Each document may include several concepts, then it is also important to be able to identify all of these concepts and quantify their presence in the document.

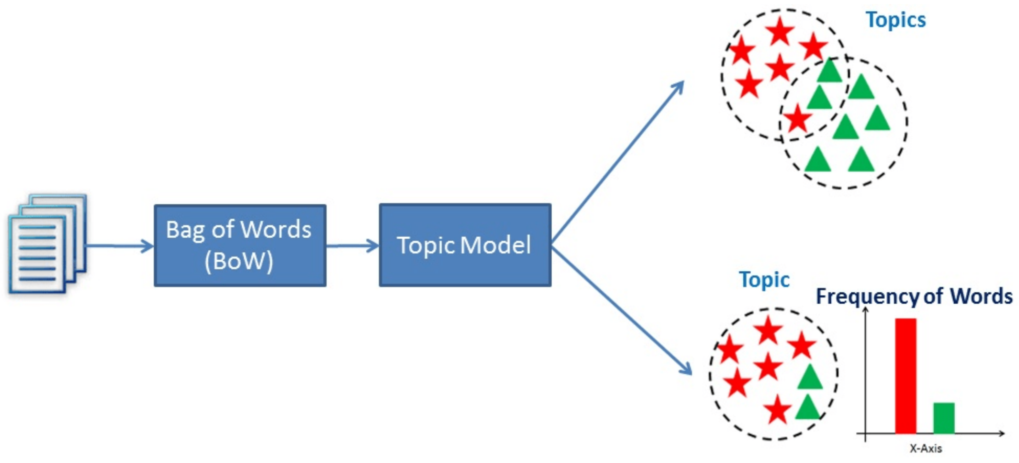


Figure 3.4: Topic Modelling process

Latent Semantic Analysis

Latent Semantic Analysis (LSA) is a method of estimating the meaning of a document, its proximity to a particular topic or subject. The method is based on the assumption that each word has a particular meaning in a particular concept. The fact that each word could have different meaning in the current language complicates the analysis.

LSA starts from the usage of the BoW representation of the documents. In fact, considering simultaneously all the words and all the documents, it is possible to derive the term-document matrix (occurrence of terms in a document), where each row is correspond to a word and each column correspond to a document. Then, LSA extrapolates the most important concepts by applying a matrix decomposition, the Singular Value Decomposition (SVD), and considering the most important terms.

In fact the is able to express the initial vectors as a combination of the most important eigenvalues and eigenvectors.

Let $A = [d_1, ..., d_N]$ our term-document matrix, where each d_i correspond to the representation of the i-th document according to the BoW model. Then A can be decomposed in the following way:

$$A = U\Sigma V^T$$

where:

- $A = M \times N$ term document matrix
- $V = N \times r$ document concept matrix, composed by the vectors v_i
- $\Sigma = r \times r$ concept weighting diagonal matrix
- $U = M \times r$ term concept matrix, composed by the vectors u_i

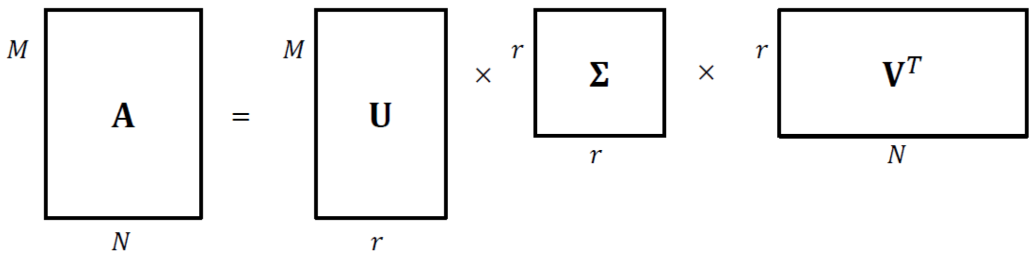


Figure 3.5: Singular Value Decomposition representation

In this way:

- r represent the number of uncorrelated concepts
- u_i represents a term-concept vector, or each element of the vector u_i represents the contribution of a word to the i-th concept

- v_i represents a documents concept vectors vector, or each element of the vector u_i represents how much a document is described by the i-th concept

Then each document d_j could be expressed in terms of the concepts space with a simple linear transformation:

$$d_{r_j} = (U\Sigma)^{-1} d_j = \Sigma^{-1} U^T d_j$$

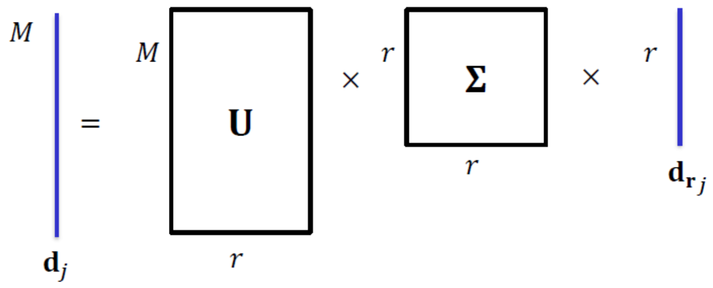


Figure 3.6: Document representation through LSA

Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation (LDA) [**5**] is one of the most famous and utilized topic model algorithm. It is based on the idea that each document can be represented by a distribution of topics and each topic can be represented by a distribution of words.

Starting from a corpus of texts, we can only observe words and documents. The concepts in this model are represented by latent variables that cannot be observed. The only thing that we can see is the relation of these topics with our corpus: how well a word belong to a particular topic, which topics each document includes.

More in detail, let's consider:

- k - Number of topics
- V - Vocabulary size
- M - Number of documents
- N - Number of words in each document

- w - Word represented as a one hot encoded vector of size V
- **w** - Document represented as the matrix composed by the N word vectors belonging to the document. Then $\mathbf{w} = \{w_1, w_2, ..., w_N\}$
- D - Corpus, that corresponds to a collection of M documents. Then $D = \{\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_M\}$
- z - A specific topic from the set of k topics. It is represented as a distribution of words.

Moreover, following the graphical representation of the LDA model in Figure 3.7:

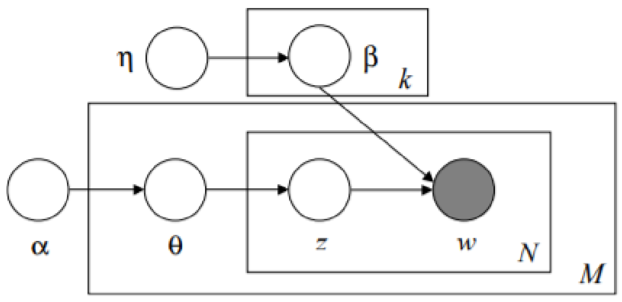


Figure 3.7: Graphical model representation of the LDA model.

- α - Distribution related parameter that governs what the distribution of topics is for all the documents in the corpus looks like
- θ - Random matrix where $\theta(i, j)$ represents the probability of the i-th document to containing the j-th topic
- η - Distribution related parameter that governs what the distribution of words in each topic looks like
- β - A random matrix where $\beta(i, j)$ represents the probability of i-th topic containing the j-th word.

The variable α is a $(M \times k)$ matrix that has a topic distribution for each document and defines θ. Then each one of the M documents has a θ distribution and has N words, where each word is generated by a topic.

η is a $(k \times V)$ matrix that has a parameter vector for each topic. Then, the variable β has a Dirichlet distribution based on η and generates k individual words for each topic.

Finally we want to find the following probability:

$$P(\theta_{1:M}, \mathbf{z}_{1:M}, \beta_{1:k} | D; \alpha_{1:M}, \eta_{1:k})$$

that corresponds to the posterior probability of θ, $\mathbf{z}$ and β, based on the corpus of documents D using the parameters α and η.

Since such posterior is not tractable, it is necessary to approximate it with some known and simpler probability distribution through the usage of the variational inference methods. In particular, a possibility is to minimize the KL divergence between the approximation and true posterior.

In a discrete probability space X, the Kullback–Leibler divergence from Q to P (both distribution in X) is defined as:

$$D_{KL}(P||Q) = \sum_{x \in X} P(x) \log \frac{P(x)}{Q(x)}$$

It measures somehow the distance between two given distribution. A lower KL divergence value corresponds to a better approximation of the desired distribution.

This brings us to the following optimisation problem:

$$\gamma^*, \phi^*, \lambda^* = \underset{\gamma,\phi,\lambda}{\operatorname{argmin}} D_{KL}\left(Q(\theta, \mathbf{z}, \beta | \gamma, \phi, \lambda) || P(\theta, \mathbf{z}, \beta | D; \alpha, \eta)\right)$$

where γ, ϕ, λ represent the free variational parameters. By changing them, it is possible to reduce the distance from Q to P obtaining a better approximation.

Number of topics

In both these methods, the number of topics to use to model the documents must be imposed before. However there exists some criteria that help in the decision of the best number of topics for a specific corpus. They are mostly based on the maximization of the likelihood or the evaluation of the topic coherence measures. The latter assumes that each generated topic has a list of words and they calculate the some features, like the average or the median, of pairwise word similarity scores of the words in a topic. The higher is the topic coherence and better should be the topic model.

In the best scenario, the representation of these measures with respect to the number of topics selected shows a unique peak, that should correspond to the optimum number of topics to be considered. In our case we considered all the possible number of topics multiple of 4 starting from 4 and arriving to 300. Unfortunately, until that point, there weren't significant peak to take into consideration. Moreover, larger numbers of topics requires much more time in the extraction of the main concepts. Even having a grater computation power, after a first qualitative analysis it was clear that each concept represented by its main words started to be meaningless and was repeated several times.

This was the main reason that make us use new sources of text. In fact we took into consideration also the answers to the queries and the Wikipedia data presented in section 1.4. The results and the measures improved, but at the end they didn't provide us clear indications about the number of topics. A larger number of topics could also influence negatively the results of the following steps and overcomplicate the problem.

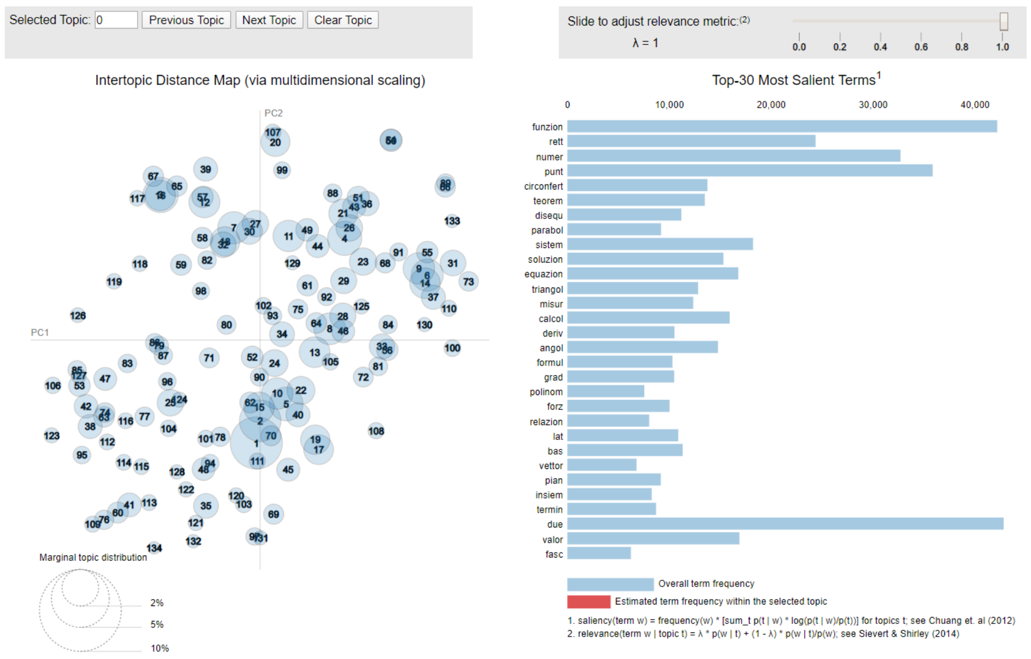
These problems made us go for a different approach. We decided to consider only the number of topics multiple of 67 (the number of the platform chapters) and select the model with the best expressive power, or the the greatest number of not repeated concept, and with the the concepts that are more human understandable. This selection was validated by the Redooc collaborators and finally we ended up with 134 topics derived from the LDA algorithm.

The Gensim library also provides interactive visualization tools that allowed us to better analyze the outcomes of the LDA algorithm. In particular, in Figure 3.8 is shown a representation of the topics in a 2 dimensional space with the t-SNE algorithm [**6**].

It is possible to notice that the topics are well distributed into the topic space and the overlaps are rare.

Then, we concentrate only on the lessons documents in order to study the presence of these topics inside them. This step is necessary to find out the topics that better represent the lessons contents. In fact, the corpus of texts we created was a mixture of many different sources, among which there are the Wikipedia pages that are not material of the platform, so they could include many other topics. Moreover, unlike the lessons, the queries and answers often include words and concepts strictly related to the problem scenario, but that affect negatively our analysis.

Since the LDA model provides the a measure of presence of the different topics within each document, we analyzed this presence in order to evaluating the most relevant topics for the platform contents. In particular we considered the average and the variance of the topic distribution through the various documents. The logarithm of these values per each topic is represented in Figure 3.9.



1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

Figure 3.8: 2D topic representation using t-SNE.

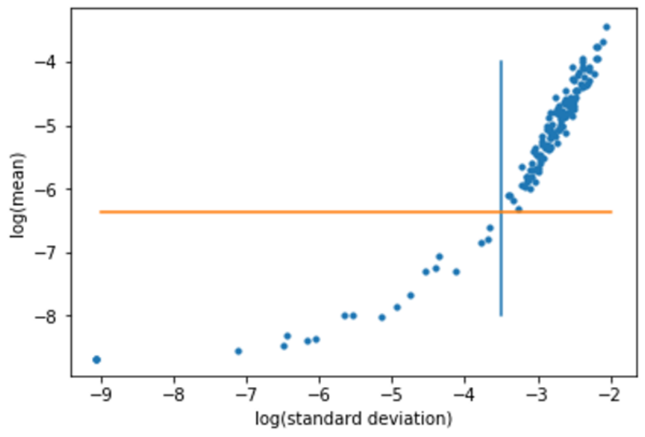
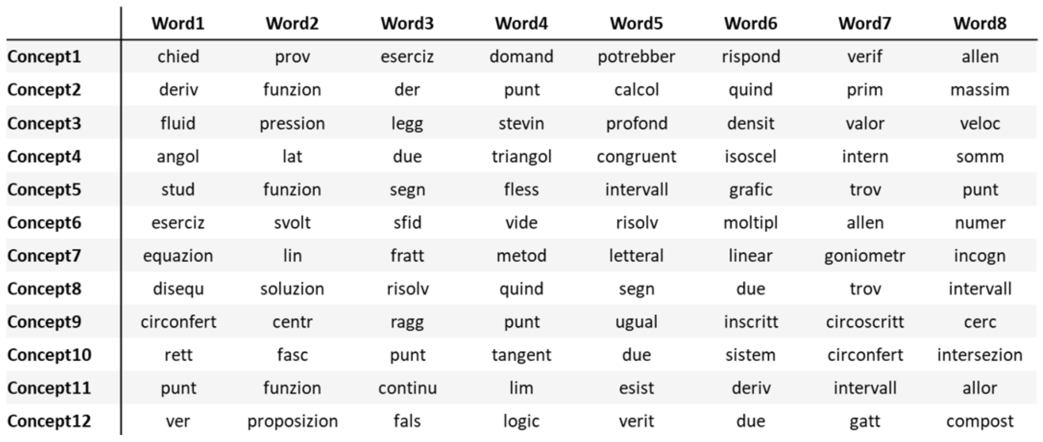


Figure 3.9: Representation of the average and the standard deviation of the presence of each topics in the different documents.

On the lower left it is possible to notice a set of topics extremely uncorrelated with the lessons material. Then, we decided to consider only the cluster of topics with a lower bound on the average and the variance, corresponding to the ones above the orange line and on the right of the blue line.

These choices led us to the final number of considered concepts, that correspond to 119 concepts. In Figure 3.10 are presented 12 of them with their 8 most significant words.



	Word1	Word2	Word3	Word4	Word5	Word6	Word7	Word8
Concept1	chied	prov	eserciz	domand	potrebber	rispond	verif	allen
Concept2	deriv	funzion	der	punt	calcol	quind	prim	massim
Concept3	fluid	pression	legg	stevin	profond	densit	valor	veloc
Concept4	angol	lat	due	triangol	congruent	isoscel	intern	somm
Concept5	stud	funzion	segn	fless	intervall	grafic	trov	punt
Concept6	eserciz	svolt	sfid	vide	risolv	moltipl	allen	numer
Concept7	equazion	lin	fratt	metod	letteral	linear	goniometr	incogn
Concept8	disequ	soluzion	risolv	quind	segn	due	trov	intervall
Concept9	circonfert	centr	ragg	punt	ugual	inscritt	circoscritt	cerc
Concept10	rett	fasc	punt	tangent	due	sistem	circonfert	intersezion
Concept11	punt	funzion	continu	lim	esist	deriv	intervall	allor
Concept12	ver	proposizion	fals	logic	verit	due	gatt	compost

Figure 3.10: Concepts representation.

These concepts represent approximations of human notions through pre-processed words. Nevertheless, it is possible to interpret them and have a domain expert assign each concept a meaning. This allowed us to evaluate the presence of these concepts in each document of the platform.

Relationships construction

The topic modelling approaches not only allowed us to find the concepts underlying the educational material, but also was able to express each educational element through the usage of such concepts. Their representation is based on distribution that the reflects the concept presence in each document. From this derives the fact that lessons that involve similar concepts should be represented by similar vectors in the topic space. Then, it is possible to evaluate somehow this distance in order to establish a degree of similarity between educational elements. These similarities will be the base of the construction of the relationships between the platform contents, one of the building blocks of the Educational Structure.

First of all, we normalized again the topic distributions among the documents, since the discard of many topics made the representation of the documents inconsistent.

Then we have taken into consideration 2 types of similarity measures:

- Cosine similarity - is a measure of similarity between two (non-zero) vectors. Since it evaluates the distance between vectors as the cosine of the angle between them, it is a metric measure of orientation and not of magnitude. Given 2 non zero real vectors a and $b$belonging to the same vectorial space, the cosine similarity could be expressed with the following formula:

$$similarity(a, b) = \frac{a \cdot b}{||a||||b||}$$

This is one of the most important used similarity measure.

- Jensen–Shannon divergence - it is method of measuring the similarity between two probability distributions. We decide of also using measures strictly correlated with the probability distributions since they are the base of the representation of the documents with the main topics. However, we didn't consider the Kullback–Leibler divergence just introduced in the previous chapters, since, even evaluating a distance between distribution, it is not symmetric. In our case the symmetry of the metric is essential since the relationships we want to evaluate are symmetric. Then we decided to consider the Jensen–Shannon divergence that represent a symmetrized and smoothed version of the Kullback–Leibler divergence. In fact, using the same notation introduced before, the Jensen–Shannon divergence (JSD) is defined as:

  $$JSD(P||Q) = \frac{1}{2}D_{KL}(P||M) + \frac{1}{2}D_{KL}(Q||M)$$

  where $M = \frac{1}{2}(P + Q)$.

Using these measures it is now possible to calculate the distance between each pair of lessons. Then, two lessons will be considered as related if the similarity measure between their distribution is grater or equal a certain threshold. The lower the threshold is and the greater the number of connections will be. Hence, the definition of such value determines the degree of association we will to maintain among the contents.

Moreover, if we consider a prerequisite relationship among the lessons, the direction of links is naturally determined by the order in which the lessons are presented on the platform. The contents on the platform follow the Italian study program, hence the base lessons are appears before of the advanced ones.

As we mentioned in section 1.1, the platform data includes also a table containing to some extent the associations among the posts and lessons of the platform. Even though it is incomplete and not constantly updated, we used it guide us in the choice of the most appropriate threshold value. In particular we considered the links of the table as correct and we evaluated the balanced accuracy of the links obtained with our method. The decision of validating with the evaluation of the balanced accuracy derived from the fact that in such a system with a large number of elements and a limited number of connection the number of non-linked elements is extremal greater than the number of the linked ones. With L lessons, the number of possible links is $\binom{L}{2}$. If each lesson is linked to other k lesson, then the number of links is $\frac{kL}{2}$ and they represent the following fraction of the total: $\frac{k}{L-1} \sim L^{-1}$. Therefore, the increment of the number of considered lessons leads to the reduction of balance between the positive case (linked lesson) and negative cases (non-linked lessons).

The balanced accuracy is defined as

$$BACC = \frac{1}{2}\left(\frac{TP}{P} + \frac{TN}{N}\right)$$

where:

- P represents the number of positive cases, or the number of linked lessons according to platform data
- N represents the number of negative cases, or the number of non-linked lessons according to platform data
- TP represents the number of cases labelled as links by both the platform data and the implemented method
- TP represents the number of cases labelled as non-links by both the platform data and the implemented method

Knowing the limitation of the used table, we did not expect high results. However we expected to find new links not already recorded.

The evaluation of the best threshold using both the types of similarity measures led us to the a balanced accuracy of 0.71 for the cosine similarity and 0.72 for the Jensen-Shannon divergence.

In order to find the relationships among the platform contents we decided to explore other possibilities of representing the corpus of text. We will describe these methods in the following section.

3.2.2 Word embedding

Word embedding refers to the set of models capable of mapping documents, phrases, or words into a vector space of real numbers, allowing an expression of their semantic and syntactic information. These models assume that words that occur in the same linguistic contexts are semantically similar and, for this reason, are represented with vectors that are close to each other.

After testing various algorithms and analyzing their outputs, we decided to use the fastText model [**7**]. It consists in a computationally efficient vector space model used for text representation, that is learning word embeddings from raw text. It consists in a two-layer neural network that takes as input a large corpus of documents and produces as output a vector space of fixed dimensions, where each word in the corpus is assigned to a vector in that space.

Even tough new very performing approaches for learning contextualized word vectors arises during the past years, like ELMo [**8**] and BERT [**9**], fastText represent the best choice in a context like ours where we have not a large amount o textual data to train the models, we have limited computational power, we don't need very complicated models since the semantic area of interest is limited to the mathematical sector, where there is a limited amount of repeated words and each of them has a specific meaning.

In its simplicity, fastText is still able to represent the syntactic and semantic meaning of each word, also showing several regularities. It is the case of male-female terms, singular-plural, nation-capital, nation-main food, ... For example the following operation among the vector representation of the indicated words $v_{King} - v_{Man} + v_{Woman}$ results in a vector very close to v_{Queen}. The distance measure used to evaluate the proximity between words is the cosine distance. The latter resulted adapted to express the constant distance representing a particular semantic meaning independent from the words considered. Each dimension of the vector representation corresponds to a specific latent meaning and the word's numerical weight on that dimension reflects the closeness of the word to that meaning.

We have tested the presence of such constant relationship among the words on our data comparing the most significant mathematical term in the singular form to their plural form. We considered the following words with the correspondent plurals: "equazione", "disequazione", "funzione", "dimensione", "numero", "metodo", "teoria", "elemento", "dimostrazione", "relazione", "soluzione", "successione", "calcolo", "teorema", "termine", "retta", "prodotto", "formula", "statistica", "insieme", "base", "reale", "tempo", "zero", "vettore", "uguale", "punto", "dato", "grado", "stato", "caso", "risultato".

A 2D representation of these words through the usage of tSNE is visible in Figure 3.11.

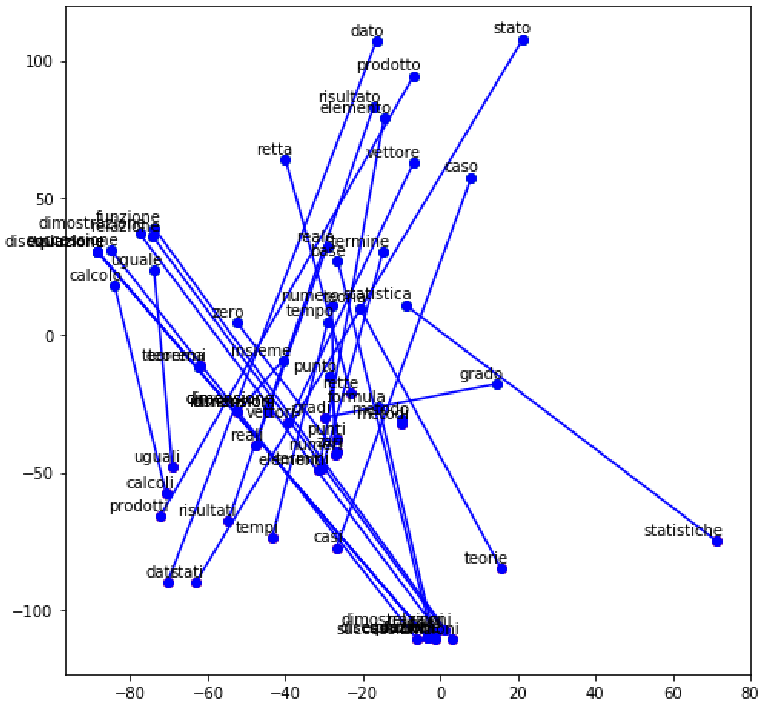


Figure 3.11: 2D representation of many significant words in their singular and plural form.

It is possible to notice that in our case there exists many different possible constants for the same type of change the semantic of the words.

The representation of all of these shades of meaning is possible only without modifying the tokenized words. Therefore, in order to keep the semantic meaning of the words it is not possible to apply preprocessing procedures like the lemmatization or the stemming. For this reason we trained the fastText model stopping the preprocessing procedures to the noise removal. For methods like BoW, Tf-Idf, LDA or Word2Vec [**10**] this aspect could represent a problem in a context where the corpus of texts is very small. Then, the representation through n-grams of fastText represent an important improvement to the Word2Vec model. This new feature in the approach allows also to evaluate the word embedding of words not present in the initial corpus.

The choice of fastText also derives from the fact that it allows a flexible representation of documents, piece of texts, phrases or set of words. In fact, from the word representation it is possible to calculate somehow the vectors representing each document. We decided to represent each document as the average of the vectors of the words that compose it. This approach summarizes the latent meanings of each document in a unique vector of the same dimension of the embedding space. In this manner it is possible to evaluate the similarity between documents or words and documents. Moreover, it guarantees the adaptability of the vector representation to the changes in the contents. In fact, the Redooc employees often changes somehow the contents of the platform. This could be due to several reason: correcting a text, splitting a lesson,adding a new educational element, deleting parts of the texts, ... The method we implemented guarantees an high level of tolerance that is not compatible with approaches like BoW, Tf-Idf, Doc2Vec [**11**], where each change requires the re training of the whole system.

Figure 3.12 shows a representation of all the lessons (red points) in 2 dimensions. The blue points correspond to the representation of the lessons indicated in the legend. We have noticed a clear clustering according to the contents and more in general to the main topics: Geometry on the right, Physics on the left, Relations and Functions in the middle and Arithmetic and Algebra ascending from the bottom.

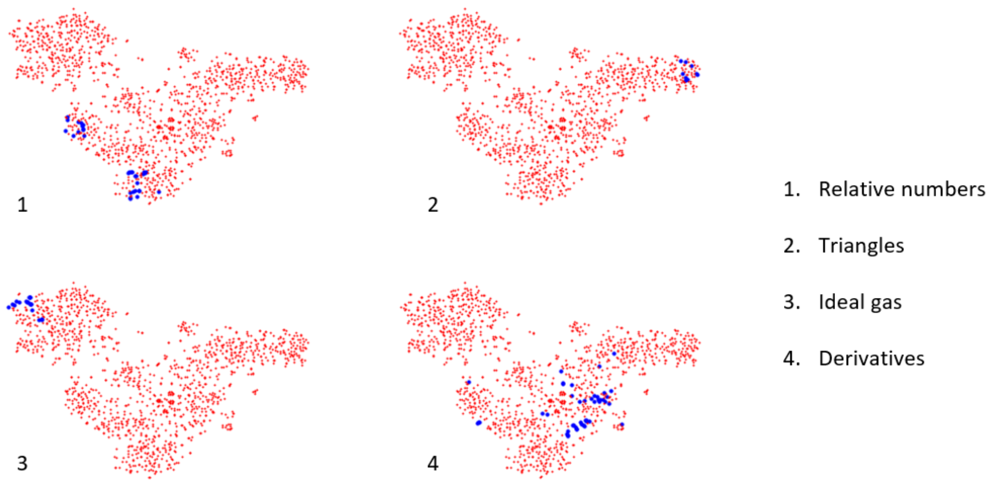


Figure 3.12: 2D representation of the lesson with specific examples.

Given the representation of each lesson, it is now possible to derive the educational structure from the definition of similarity as in the previous approach.

Since the word embedding does not represent probability distributions, we only considered the threshold on the cosine similarity obtaining a balanced

Topic	Physics	Rel. and Func.	Arit. and Alg.
Geometry	Trigonometric func.	Measures theory	Cartesian plane
Physics	/	Deriv., Integrals	/
Rel. and Func.	/	/	Polynomial func.

Table 3.1: Interdisciplinary lessons.

accuracy of 0.74.

3.2.3 Final Knowledge Structure

In order to improve the Knowledge Structure and take into account both words frequencies as well as word semantics, we decided to create a final knowledge structure by combing the output of the Latent Dirichlet Allocation model (LDA) with the output of the fastText model.

In order to be consistent and thanks to the fact that the results obtained in section 3.2.1 are very similar, we considered the cosine similarity in both the approaches. Then we considered the links that are identified by at least one of the methods when varying the thresholds.

Hence, by joining the relations of the two models and refining the thresholds we obtained a balanced accuracy of 0.79. Then, the new structure is capable of defining more accurately the prerequisite relationship between educational elements.

Figure 3.13 shows a representation of the Knowledge Structure like an oriented graph where each node corresponds to a lesson and each link correspond to a prerequisite relationship. The subdivision in topics is represented with different colours. The analysis of the graph contents and links highlights the fact that the the lessons between two topics are are somehow ”interdisciplinary” (Table 3.1).

3.3 Knowledge modelling

In order to trace a student's knowledge, we introduced, with the weighted overlay model in section 2.2, vectors of probabilities representing the student's level of knowledge related to each concept. This allows us to represent the level of comprehension of a concept through a model that reflects the probability of the user of knowing a particular concept.

However, in order to fully understand a concept, it is often necessary to master its different meanings or interpretations. In fact, from the analysis of

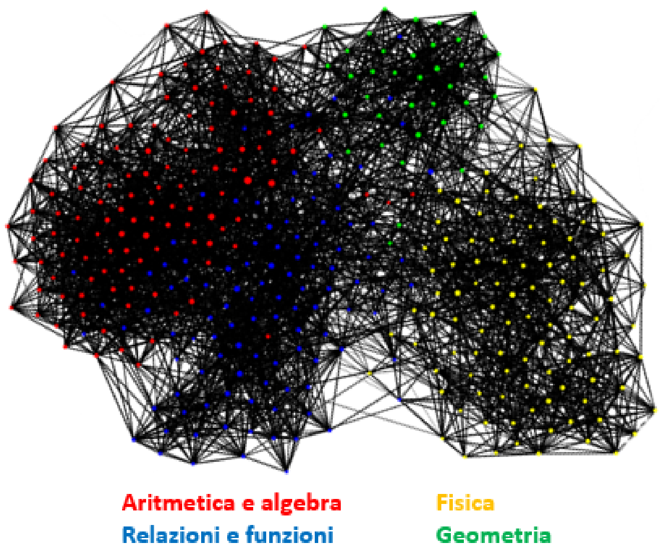
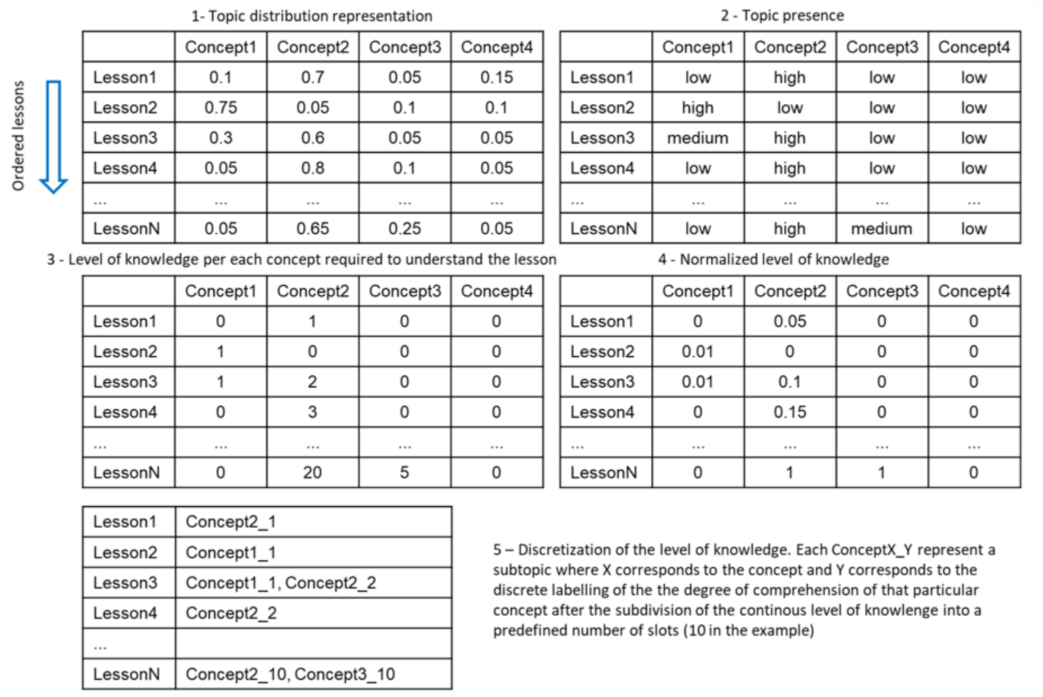


Figure 3.13: Knowledge Structure

the distribution of concepts inside each document, it is apparent that many documents share the same concepts even though they belong to different chapters, topics or school years. An example is the concept related to the trigonometry: it can be found in lessons about Geometry, where there is the geometric definition of sine, cosine, etc., and lessons about Relations and Functions, where we can find the derivatives and integrals of the trigonometric functions. Therefore, it is possible to have a certain comprehension of the geometric meaning of trigonometry and a different comprehension of how trigonometry is used in calculus. We thus decided to go more into detail and introduce subtopics, which model the different levels of comprehension of a concept.

To do this, we evaluated the level of concept comprehension that each lesson provides based on the presence of the concepts and the lesson order. Given a specific lesson, if a concept has a large presence then a new aspect of it is probably introduced and a student's level of comprehension of the concept should increase. Otherwise, if the concept has a small presence, then it is probably just mentioned and the level of comprehension does not increase. In a similar way we expect that a concept treated further along the school career requires a better understanding of the topic. To mathematically represent the different levels of comprehension of a concept, we split the initially unique level of comprehension into a fixed number of intervals, that we will call subtopics. We then assigned to each lesson the corresponding level of knowledge of each subtopic. A numerical example of the previous process is shown in Figure 3.14.



1- Topic distribution representation

	Concept1	Concept2	Concept3	Concept4
Lesson1	0.1	0.7	0.05	0.15
Lesson2	0.75	0.05	0.1	0.1
Lesson3	0.3	0.6	0.05	0.05
Lesson4	0.05	0.8	0.1	0.05
...	...	...	...	...
LessonN	0.05	0.65	0.25	0.05

Ordered lessons

2 - Topic presence

	Concept1	Concept2	Concept3	Concept4
Lesson1	low	high	low	low
Lesson2	high	low	low	low
Lesson3	medium	high	low	low
Lesson4	low	high	low	low
...	...	...	...	...
LessonN	low	high	medium	low

3 - Level of knowledge per each concept required to understand the lesson

	Concept1	Concept2	Concept3	Concept4
Lesson1	0	1	0	0
Lesson2	1	0	0	0
Lesson3	1	2	0	0
Lesson4	0	3	0	0
...	...	...	...	...
LessonN	0	20	5	0

4 - Normalized level of knowledge

	Concept1	Concept2	Concept3	Concept4
Lesson1	0	0.05	0	0
Lesson2	0.01	0	0	0
Lesson3	0.01	0.1	0	0
Lesson4	0	0.15	0	0
...	...	...	...	...
LessonN	0	1	1	0

Lesson1	Concept2_1
Lesson2	Concept1_1
Lesson3	Concept1_1, Concept2_2
Lesson4	Concept2_2
...	
LessonN	Concept2_10, Concept3_10

5 – Discretization of the level of knowledge. Each ConceptX_Y represent a subtopic where X corresponds to the concept and Y corresponds to the discrete labelling of the the degree of comprehension of that particular concept after the subdivision of the continous level of knowlenge into a predefined number of slots (10 in the example)

Figure 3.14: Numerical example of the steps involved into the creation of the subtopic labelling

In this way we were able to express each lesson as a combination of subtopics that consist in a refinement of the previous concepts. Then, these subtopics allowed us to better model the level of knowledge of each user as we will see in Chapter 3.

3.4 Query-Lesson association

In traditional educational methods, in order to understand the level of preparation of each student, teachers test them using exams. Similarly, in order to trace the level of knowledge of each student, we decided to analyze the exams associated to each lesson. In particular, we focused on the knowledge that a student should have to correctly answer a specific query. By assuming that a student able to correctly answer a query knows the concepts related to that query, it becomes important to understand the concept/subtopic distribution of each query.

Since queries tend to be short and each query could require different topics from different lessons, it is not possible to infer the subtopic distribution directly from the text of the query. For this reason, we utilized the already calculated words embedding to represent each query as the vector correspondent to the vector representation of the associated answers. We used the text of the answers for several reasons: the size of the queries is in general much smaller than the size of the answers, the explanation in the answer includes all the key words related to concepts needed to solve the problem, the words used to formulate the problem are sometimes useless, misleading or not inherent to the mathematical context. Then we obtained a representation of the queries in the same vector space we utilized to represent the lessons. This allowed us to calculate the similarity between lessons and queries and to associate each query to the most similar lessons. Hence, it was possible to approximate the subtopic distribution of each query utilizing the subtopic distribution of the most similar lessons. The similar lessons were filtered using the same threshold used in section 3.2.3 associated to the fastText approach. In order to guarantee that each query would be related to at least one lesson, we associated by default to each one of them the subtopics present in the lesson containing the level the query belongs to.

Finally, we counted per each query the number of times each subtopic has been associated to it. The subtopics with a frequency lower than the average were discarded and the remaining were associated definitively to the query. In this way, we kept all the subtopics that could be considered as more "useful" in order to answer the question. All the queries were labelled using this method, creating the same structure obtained for the lessons (last step in the labelling procedure in Figure 3.14)

Chapter 4

User profiling

In order to offer the best personalized learning experience, it is necessary to know the users as well as possible. For this reason, many of the web-based educational systems often require the completion of questioners and surveys at the time of registration or during the usage of the platform. The questions may be related to social information, like age and gender, as well as general information, like interests and goals. Moreover, some platforms propose entry tests to determine the initial level of knowledges of the user. All this information contributes to the creation of a user profile, that is a mathematical model that summarize all the useful features of the user. The user profile is continuously updated based on the data gathered on the platform, such as exam results, navigation history, etc.

In our case, we have limited user information (see Chapter 1.2) and therefore must create the user profile only based on the user's level of knowledge, which can be derived from the analysis of the exams results (see Chapter 1.3). The user profile, initially unstable, becomes more and more accurate over time with the extended usage of the platform and specifically the number of exams taken. Furthermore, the development of a model able to profile and trace the knowledge of the students is required to effectively create an adaptive learning system capable of recommending the best educational material. The recommendation must be based on the level of knowledge of the user and help them obtain a complete preparation of every concept, filling knowledge gaps.

In Chapter 3 we laid the foundation for the creation of a framework that allows us to trace the level of knowledge of each student. As mentioned in section 3.3, it is possible to model the students through the usage of vectors that store estimates of their level of comprehension of each concept. This kind of model is known as an overlay model and represent one of the most common user profiling models. These models generally inherit the concepts, or in our case the subtopics, from the knowledge structure.

Therefore, by expressing these estimations of their level of knowledge as the probability of knowing each particular subtopic, we can profile each user and trace their changes over time through the use of knowledge tracing algorithms. They model learning in time and tries to predict the performance of the student in future tests. Based on these predictions, it is then possible to elaborate strategies to optimize the learning experience.

During the years many different algorithms have been developed. The most important ones are ([**13**], [**12**]) the following:

- Item Response Theory (IRT);
- Bayesian Knowledge Tracing (BKT);
- Performance Factor Analysis (PFA);
- Deep Knowledge Tracing (DKT).

Item Response Theory

Item Response Theory (IRT) [**15**] was one of the first tracing algorithms implemented. It is based on the idea that the students' performances during the tests (or a general task) depends on the level of mastery of latent unobserved variables, or the students' "skills". In psychometrics this relation is represented by the cumulative distribution function of the normal distribution, that can be approximated with a logistic distribution. Hence, in its simplest and most used formulation, the IRT estimates the probability that the student i answers the query j correctly as:

$$p(\theta_{ji}) = \frac{1}{1 + e^{-(\theta_{ji} - \beta_j)}}$$

where θ_{ji} represents the level of mastery of the student i in the skill θ needed to answer the query j and β_j corresponds the general level of difficulty of the the query j. Then, given the responses of the students, it is possible to infer their knowledge by using expectation maximization algorithms.

The IRT model is characterized by 2 limitations: it assumes that one single skill is involved in each problem and that the knowledge is static during the entire test. A "static" knowledge means that the students have not the possibility of increasing or decreasing their proficiency in the various skills during the duration of the test in the learning environment they are exposed to.

Bayesian Knowledge Tracing

In a context where the users are continuously tested with exams, the Bayesian Knowledge Tracing (BKT) algorithm represents one of the most common models used to track the process of student knowledge acquisition. In fact, it is a user profiling method used in many tutoring systems to model each learner's mastery of the knowledge being imparted.

In its standard implementation, BKT assumes that each possible user skill can be modelled as latent binary variables (learned vs not learned) and learning is characterized as a transition between these two states. Therefore, a student's knowledge can be represented as the set of all these binary variables, one per skill, where the skill is either mastered or not. Input observations in BKT also tend to be binary: a student gets a problem either right or wrong. The standard BKT model is based on 4 parameters (see Figure 4.1):

1. Initial learning probability $P(L_0)$ – the probability a concept is already in the learned state prior to the first opportunity to apply the skill;

2. Acquisition probability $P(T)$ – the probability a skill will make the transition from the not learned to the learned state following an opportunity to apply the skill;

3. Guess probability $P(G)$ – the probability a student will correctly answer a problem without knowing the related skill;

4. Slip probability $P(S)$ – the probability a student makes a mistake applying a known skill.

Utilizing these parameters, it is possible to infer the actual probability $P(L)$ of every student of knowing a particular skill after the application of a it in a problem.

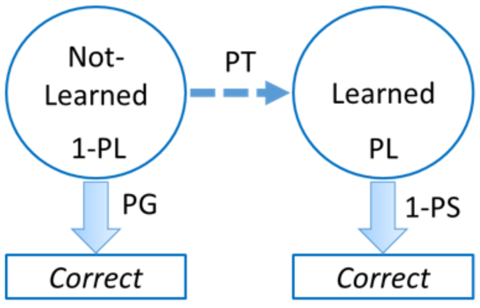


Figure 4.1: Representation of the Bayesian Knowledge Tracing model

Assuming that these parameters are set for all skills, let indicate with $P(L_n^c)$ the probability that a concept c is learned by a student at the n-th opportunity of applying it. BKT represent a Hidden Markov Model (HMM) where each state is update using the Bayes' Theorem:

$$P(L_{n+1}^c) = P(L_n^c | Action_{n+1}) + (1 - P(L_n^c | Action_{n+1}))\, P(T^c)$$

where

$$P(L_n^c | Action_{n+1} = Correct) = \frac{P(L_n^c)(1 - P(S))}{P(L_n^c)(1 - P(S)) + (1 - P(L_n^c))P(G)}$$

$$P(L_n^c | Action_{n+1} = Incorrect) = \frac{P(L_n^c)P(S)}{P(L_n^c)P(S) + (1 - P(L_n^c))(1 - P(G))}$$

The prediction of the performance of the student is given by:

$$P(Action_{n+1} = Correct) = P(L_n^c)(1 - P(S)) + (1 - P(L_n^c))P(G)$$

Many methods are used to establish the best parameters, as expectation maximization algorithms, conjugate gradient search and grid search.

The BKT algorithm allows the modelling of the users in a learning environment where they can continuously increase their knowledge each time they use a skill. However, it is based on the idea that knowing the applied concepts generally leads to correct answers and, vice versa, correct answers implies that a student knows the relevant skill. Then, it is important to not create degenerate models and continuously ensure that correct answers result in an increase of the probabilities of knowing the concepts involved and incorrect answers lead to their decrease.

Even if the learning is not considered as static, the model is still characterized by the limitation of associating a single skill to each query.

Performance Factor Analysis

Performance Factor Analysis (PFA) [**16**] corresponds to an elaboration of the IRT and an alternative to the BKT algorithm. This model is able to take into consideration the application of multiple skills during each query through the introduction of of a new value m representing the accumulated learning for student. The latter is defined as follow:

$$m(i, j, c \in C, n) = \beta_j + \sum_{c \in C} (\gamma_c s_{i,c} + \rho f_{i,c})$$

where

- i, j correspond respectively to the i-th student and the j-th query
- c is one of the concepts C applied to solve the problem
- β capture the easiness of the concepts, s and f track respectively the prior successes and failures of the student in applying the concepts C, and γ, ρ corresponds are variables responsible of scaling the effect of these observation counts

Then, the performance predictions are modelled as a logistic function of the new variable introduced:

$$p(m) = \frac{1}{1 + e^{-m}}$$

Finally, the parameters β, γ, ρ are fitted to maximize likelihood of the model according to the data.

Deep Knowledge Tracing

The recent interest in Machine Learning and Deep Learning have lead during the last decade to the growth of new tracing models based, or partially based, on Neural Networks. In [**17**] the authors explore the utility of using Recurrent Neural Networks (RNNs) to model student learning, introducing the formulation of the Deep Knowledge Tracing (DKT). The latter is based on a Long short-term memory (LSTM) architecture where the student's knowledge is dinamically represented in a latent space and inferred by the past history, or the previous performances.

In its simplest formulation, the student interactions, that are the combination of which query is answered and which is the outcome, correspond to the input of the model and are represented with one-hot encoding vectors $x_t \in \{0,1\}^{2M}$, where M is the number of unique exercises in the dataset. Then, the predicted probabilities of answering correctly are represented by M-dimensional vectors defined by the following equations:

$$h_t = \tanh\left(W_{hx}x_t + W_{hh}h_{t-1} + b_h\right)$$

$$y_t = \sigma(W_{yh}h_t + b_y)$$

where

- W_{hx}, W_{hh}, W_{yh} are respectively the an input weight matrix, recurrent weight matrix and readout weight matrix;
- h_t can be considered as encodings of relevant information;

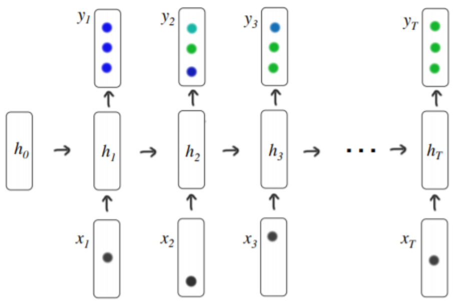


Figure 4.2: Representation of a simple Recurrent Neural Network

- b_h, b_y correspond to the biases for latent and readout units.

The training follows the maximization of the likelihood of according to the sequence of the answers given by the users.

Even if the architecture based on LSTMs latent units increases the complexity of the model with respect to the other previously presented or to the classical RNN, its particular structure makes it ideal for the kind of problem we are considering, where latent skills are involved in the completion of tests somehow comparable to time series. In fact, the DKT outperformed all the previous algorithms in the task of predicting the future users performances.

one of the most famous knowledge tracing algorithms, Bayesian Knowledge Tracing. We overcame the limitations of the standard version of the algorithm by adapting it to an environment where multiple skills per each query are allowed. Finally, we combined this model with a Neural Network to be able to empirically evaluate variables. This final model, denoted as BKT+NN model, was tested and validated on the period from October 2017 to July 2019.

4.1 Alternative to traditional Bayesian Knowledge Tracing

During the years many different knowledge tracing algorithms has been implemented and they often involve one or more different versions of the models just presented. Moreover, the application of standard knowledge tracing algorithms is typically based on tests in a controlled environment where the same (per each student) limited number of exercises and concepts are involved and each query has been manually labelled with its unique correct skill. In this framework, the application of the tracing models and the analysis of the outcomes results easier and immediate.

In our case, we have to implement a system with the following relevant characteristics:

- not invasive - the new system does not to impact the actual structure and functioning of the platform and has to involve as less modifications as possible;
- able to detect the lack of knowledge - one of the primary goals of the project is to create a system able to detect in a dynamic learning environment the level of knowledge of the student in each topic and, in particular, the lack of knowledge in the various arguments;
- recommender - the system have to recommend the best lessons in order to optimize the learning. In particular it has to be able to propose the best lesson to fill the various gaps in the knowledge and guarantee a solid education in any field;
- simple and fast - a certain degree of simplicity has bee required in order to limit the requirement of additional resources (computational power and memory space) to execute the codes. Moreover, the interaction with the users needs to be almost immediate each time the algorithms analyze the data and elaborate the best recommendations to propose;

- manageable - the system has to be somehow understandable by the managers of the platform so that they will be able in the future to change the various parts of the implemented models in order to update the entire software.

In order to respect all these requirements, we decided to elaborate a modified version of the BKT algorithm that traces the level of knowledge of each student in a learning environment where:

- there is no previous information about the student and the algorithms are completely based on the data relative to the results of each student in the various queries;
- the Knowledge Structure has not been previously provided and it has inferred by the analysis of the texts of the platform as indicated in chapter 3;
- there is the possibility of having multiple subtopics in the same query. The procedures conducted to create the Knowledge Structure are based on the fact that the same words appears in different topics and are related to different level of comprehension. From here we have the introduction of the subtopics capable of creating a detailed representation of the knowledge covered on the platform;
- there exists an underlying correlation between the subtopics that depends on the subtopics themselves or on the way the user have learned them. In the first case the subtopics could share some common meaning (for example the trigonometric functions and the triangles) or they could simply be a different expression of the same topic, but with a different level of knowledge. In the second case, the fact that a user have or not have learned a certain topics and the way and the order in which he/she learned them could influence the knowledge and the perception of other subtopics;
- the presence of a large number of queries and subtopics and the fact that each student ha a singular atypical behaviour have forced us to consider models with great adaptability and able to continuously deal with the cold start problem. In fact, each user tries in general different levels and a different order, with large periods of time between one usage of the platform and the other. In this framework, each time a student connects to the platform could be almost considered as a new user, even if we have already information about him/her. In these circumstances, all the possible recursive patterns are minimal and limited to few exams, making any classical inference of the model parameters pointless;

- after answering each query there is an explanation of the solution. The letter affect the level of knowledge of the student and should be considered as a source of learning.

In order to take into consideration and overcome all these problems, we introduced an advanced version of the BKT algorithm that also incorporates the possibility of having multiple subtopics in the same query.

4.1.1 Bayesian Knowledge Tracing with multiple skills correlation (BKTMS)

Let consider a generic query where a set C of different concepts c are involved. Then, the probability of knowing the concept c at time n is indicated as $P(L_n^c)$. Notice that the time $n \in \mathbb{N}$ could not necessarily correspond to the n-th opportunity of applying the concept c. In fact, for each concept $c' \notin C$ that has not been involved in the resolution of a particular problem at time n, it applies $P(L_{n+1}^{c'}) = P(L_n^{c'})$. Moreover, let $P(L_n^C) = P(\bigwedge_{c \in C} L_n^c)$ and $P(L_n^{-c'}) = P(\bigwedge_{c \in C \setminus c'} L_n^c)$ respectively the probability of having learnt all the subtopics in C and the probability of having learnt all the subtopics in C without considering a particular concept c'.

The classical probabilities $P(T)$, $P(G)$, $P(S)$ are defined as usual and are specific per each query. Hence, with these parameters it is possible to infer the actual probability of knowing a particular concept after its application using again the Bayes' Theorem:

$$P(L_{n+1}^c) = P(L_n^c|Action_{n+1}) + (1 - P(L_n^c|Action_{n+1}))\, P(T)$$

where

$$P(L_n^c|Action_{n+1} = Correct) = \frac{P(L_n^c)\,[P(L_n^{-c})(1 - P(S)) + (1 - P(L_n^{-c}))P(G)]}{P(L_n^C)(1 - P(S)) + (1 - P(L_n^C))P(G)}$$

$$P(L_n^c|Action_{n+1} = Incorrect) = \frac{P(L_n^c)\,[P(L_n^{-c})P(S) + (1 - P(L_n^{-c}))(1 - P(G))]}{P(L_n^C)P(S) + (1 - P(L_n^C))(1 - P(G))}$$

The prediction of the performance of the student is given by:

$$P(Action_{n+1} = Correct) = P(L_n^C)(1 - P(S)) + (1 - P(L_n^C))P(G)$$

Notice that the new model keeps into account the possibility that the outcome of the problem could be influenced by subtopics different from the one considered. In fact, it is sufficient to not master a singular concept to give an incorrect answer. However, in this formulation it also essential to know the probabilities $P(L_n^C)$ and $P(L_n^{-c'})$. They represent the probabilities of knowing all or all except for 1 the concepts involved in the query. In general they can not be expressed as a simple product of probabilities since, as mentioned before, the student's knowledge of a particular concepts could be related to the knowledge of one other. In general $\prod_{k\in K} P(L_n^k) \leq P(L_n^K) = P(\bigwedge_{k\in K} L_n^k) \leq \min_{k\in K} P(L_n^k)$, but these relationships still represent too wide constraints. In fact, in the simple case where $P(L_n^k) \approx P(L_n^{k'}) \approx p$ for each $k, k' \in K$, we obtain $p^{|K|} \leq P(L_n^K) \leq p$, with $p^{|K|}$ exponentially decreasing to 0 with the increase of the number of concepts involved. Unfortunately there is not a way of calculating precisely such values since they are based on the correlations among the concepts that also depend on factors external to the platform and related to the student. Nevertheless, during our analysis we tried to estimate these probabilities in many different ways in order to maximizing the prediction accuracy of the algorithm. In particular we assumed that in general they could be approximated with the following function:

$$P(L_n^K) = (1-\lambda)\prod_{k\in K} P(L_n^k) + \lambda \min_{k\in K} P(L_n^k)$$

where λ is a real value between 0 and 1 that somehow summarizes correlation among the concepts. We expressed it as a constant or as a function of the number of concepts involved $\lambda = \lambda(|K|)$. In the second case we assumed that a sort of inverse relationship exists between λ and $|K|$, so that with the increase of the number of concepts, there will be also an increase of the factor of $\prod_{k\in K} P(L_n^k)$ that leads to a diminution of the general probability $P(L_n^K)$. In fact, it is reasonable to assume that more concepts implies more correlation. Hence, we expressed λ as $\lambda(x) = \frac{1}{x}$ and $\lambda(x) = 2^{-x+1}$, keeping in mind that all the functions tried have to respect known relationship when only one single concept is involved: $\lambda(1) = 1$.

Finally, we tested all these methods with different parameters in order to find the best model in terms of accuracy in predicting the future performance of the users.

4.1.2 Bayesian Knowledge Tracing with Neural Network predictions (BKTNN)

The algorithm just introduced is able to predict the user's knowledge starting from the BKT parameters defined at the the beginning of chapter 4 as well as a new correlation variable. The latter has the role of summarizing the relationship between the different subtopics each time the user utilizes one of them. Even though this new method allows us to trace the users' level of knowledge in presence of multiple skills, it also raises the issue of evaluating the new function introduced. In fact, in this case the correlation function involve a multitude of different factors, from the correlation amongst the subtopics themselves to the evaluation of the external unknown elements related to the student's characteristics and history (mainly the order and the way they learned the concepts). All these factors make it impossible to estimate the variables thorough traditional methods. The outcomes of the conducted tests reflected these various problems. The introduction of a new variable independent from the probability rules led to the inconsistency of the model during some iterations:

- the definition of $P(L_n^K)$ as a function of the λ variable does not always guarantee the basic probability constraints $0 \leq P(L_n^k) \leq 1$. After a certain number of iterations where the student successfully demonstrated the mastery of the same concepts simultaneously used in different queries, the model can provide an estimation of the new probability $P(L_{n+1}^k)$ of knowing a concept k greater than 1. In these cases we manually imposed $P(L_{n+1}^k) = 0.99$ in order to be able to continue with the knowledge tracing;
- there exists a combination of variables that lead to the degeneracy of the model. In fact, we have seen that certain intervals of the probability $P(L_n^k)$ and certain values of λ result in a decrease of the estimation of $P(L_{n+1}^k)$ with correct answers and an increase of it with incorrect answers. This violates the basic assumptions of the BKT model.

Then, we decided to combine the advanced version of the BKT described in section 4.1.1 with a neural network that, receiving as input a selection of features derived from the user's skill and history and the characteristics of the query, is able to predict the result of the query and provide an evaluation of the correlation parameters. Utilizing this state-of-the-art method, which we shall indicate as BKTNN model, we are able to trace the students' knowledge.

We tried several different combination of data administration, features extraction and neural network structures in order to obtain the best performances. Then, the best combination has been used to construct the final model architecture and to train the algorithm. We will present it in the following section in conjunction with the achieved results.

4.2 Model training

Since we are utilizing Machine Learning algorithms, we had to decide how to manage the data and in particular on which period to train our models. In

Chapter 1 we had analyzed the user information and exam material, noticing that:

- There is a seasonal behavior of the new subscriptions, with peaks during the months of October and January (see section 1.2);
- There is a seasonal behavior of the completed exams, with a pattern that mirrors the subscriptions;
- The peak in completed exams in correspondence to the peak in subscriptions implies that the majority of the student utilize the platform only for a short period of time after subscribing. In fact, we discovered a significant churn rate: the average number of completed exams per student related to the educational material we considered is 8;
- The high churn rate, coupled with the fact that the school program may change from year to year, implies that older data become obsolete and less reliable.

We therefore decided to train the BKTNN model on all the exams completed during the previous scholastic year. Instead of building specific models per one or more specific exams or students, we constructed a unique model on all the exams able to understand both aspects related to the individual student, such as their skills, as well as the similarity between exams and users. Then, by training on the most recent year of data, we allow the model to understand an educational contest that is as updated as possible.

The framework in which the data are collected is particularly suitable for an analysis that treats the students' sequences of queries as a time series. A possible implementation could consider the queries' information and the users' feature as the input data. In particular, the users' feature would include variables that summarizes the users' history alongside latent variables regarding the users' knowledge. Finally, the latter would be determined according to the binary outcomes of the queries at each discrete time step. Such approach is very interesting and partially reflects the cutting-edges methods [**17**]. However, unfortunately, for the reasons explained in section 4.1, the data in our possession don't respect the minimum of regularity and availability to implement and train algorithms based on such methodologies. The main problems regards the huge diversification in the attempted exams, their order and the lack of information about the users' between a session on the platform and the other. The difference in the students' knowledge between a session and the other are unpredictable and implies continue gaps in a possible implementation based on time series.

Our particular contest forced us to elaborate different and more general solutions focusing more on the available data. The greatest benefit of our approach is to be able to infer the knowledge of the users without having initial information about them and in a dynamic environment. In fact, many knowledge tracing algorithms are used in a controlled environment where a specific test is proposed to a group of students and the algorithm must predict the results based on a preliminary test and the live performances during the test. In these conditions, the student knowledge does not vary much and therefore the prediction task is made easier. In our case, we are able to trace the students' skills over a long period of time, in a situation where the level of knowledge is dynamic and variable.

4.2.1 Backtesting

In order to evaluate how well the BKTNN model "understands" the level of knowledge of the students, we decided to evaluate its performances in terms of the accuracy in predicting the queries results. In particular we carried out a simulation on the period between October 2017 and September 2019, where there is a more representative and more update part of the data.

Considering the particular structure of our models, we decided to use a validation technique called Moving Window Cross Validation (MWCV). It is used for time series and sequential data and essentially consists in using past data to make predictions in the future. Given a dataset with a seasonality and a selected period, the model is trained on the data belonging to the past and then tested on the observations belonging to the considered period. The procedure is then repeated for a rolling period of time. In particular we chose to train the model not on all past data but only on a fixed window, that is for a limited number of periods into the past. A representation of the rolling-origin MWCV is represented in Figure 4.3.

Specifically, we split each scholastic year into sets of 4 month ("quadrimestre"), starting from October, February and June in order to follow the classic trend of the Italian scholastic system. This choice reflects the considerations made in section 1.3 and at the beginning of section 4.2. After creating this time subdivision, we tested the model on a certain "quadrimestre" after training it on the previous year (therefore 3 "quadrimestri").

Besides the BKTNN model, we constructed a "baseline" model to be used as a comparison with our model. Following the time subdivision presented, the baseline model estimates the outcome of each query of each student in a particular period as the majority of the outcomes of the same query calculated on the previous 3 "quadrimestri".

The execution of the BKT algorithm follows the scholar year subdivision,

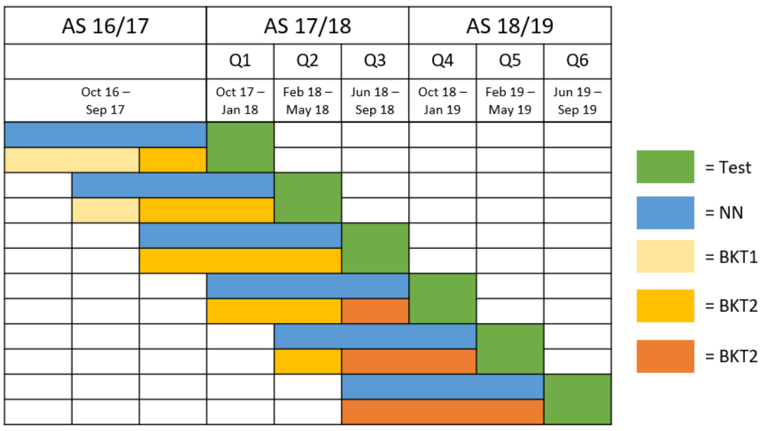


Figure 4.3: Rolling-origing MWCV

or it starts in October and ends in September of the following year without interruptions. Hence in many cases data belonging to different BKT processes were used to train the neural network.

Before training the final network, it is necessary to initialize and execute the BKT algorithm in order to get the features needed in the refinements of the predictions. The various BKT parameters were initialized as follow:

- $P(G)$ corresponds to the probability of guessing without knowing the answer. Then, following its definition, we assigned the probabilities 0.25, 0.17 ($\approx \frac{1}{6}$), 0.5 respectively to the single choice, multiple choice and true-false queries. They reflect the probabilities of randomly choosing the correct answer among the proposed ones;

- $P(L_0^c)$ corresponds to the probability of knowing the concept c prior to the first opportunity of applying it. Then, we assigned this value the baseline probability of the baseline model introduced in the previous paragrapher. In this manner, each time an user applies a new concept in a query, the probability of knowing it is associated with the best estimation of the probability of a correct answer based on the past behaviours of the students in that particular query;

- $P(S)$ and $P(T)$ has been assumed constant and estimated with a brute force approach [**18**]. A grid search with different values between 0.1 and 0.3 has been carried out for both the variables in order to maximize the general accuracy. Like for $P(L_0^c)$ and $P(L_n^K)$, also these probabilities derive from the combination of information regarding the queries themselves and the individualized students histories. Hence the assumption of constant values in order to simplify the model and allow a simpler future update and management of the system.

Then, with this configuration the predicted probabilities were used to train the neural network as further input data. We trained a fully connected neural network using the dropout [**23**] and early stopping approaches [**24**].

4.2.2 Results

We have simulated the behaviour of the various models and evaluated their performances over the period October 2017 – July 2019, where the percentage of correct answers is 58%. We discovered that the use of the BKTNN model represents a significant improvement (see Figure 4.4):

- the baseline has an average accuracy of 64%. We have also subdivided the dataset into different regular time periods and recalculated the various percentages. We obtained similar percentage in all of these different splits, verifying the consistency of the method;

- the BKTMS model achieves the 62% of accuracy in its best configuration where $P(S) = 0.15$, $P(T) = 0.2$ and $\lambda = \frac{1}{|K|}$. The fact that the probability $P(S)$ does not respect the constraints proposed in [**14**] could reflect the inconsistency in the users' answer in queries regarding the same topics where the same concepts are involved (we will explain this point later in the section). Even if the model performances are worse than the baseline ones, the BKTMS can still represent a source of new features for the neural network model and a way to trace the students' knowledge;

- the BKTNN model is able to predict the outcome of the queries with an accuracy of 73%.

  In its best configuration it takes as input per each student the previous average of correct answers, the baseline probability for the query based on the previous 3 quadrimestri, the number of concepts involved, the prediction $P(Action_{n+1} = Correct)$ of correctness evaluated by the BKTMS model, the mean, maximum, minimum and standard deviation of the updated probabilities $P(L^c_{n+1})$ of knowing the concepts involved, the mean, maximum, minimum and standard deviation of the percentages of correctness of the student per each concepts involved calculated as the percentage of times per each concepts he/she has answered correctly to the questions where that particular concept is applied, and finally several binary dummy variables that report the actual presence of the previous variables or the use of default values.

The fully connected neural network is composed by 4 hidden layers with respectively 50, 100, 50, 10 neurons and an output layer of one neuron ReLu . The activation functions of the nodes of hidden layers and the node of the output layer are respectively the ReLU function and the sigmoid function. The usage of the sigmoid function also in the hidden layers slightly worsens the performances. The loss function is the binary cross entropy according to the data characteristics. The optimizer used is the Adam optimizer [**19**] with the default Keras parameters (https://keras.io/optimizers/).

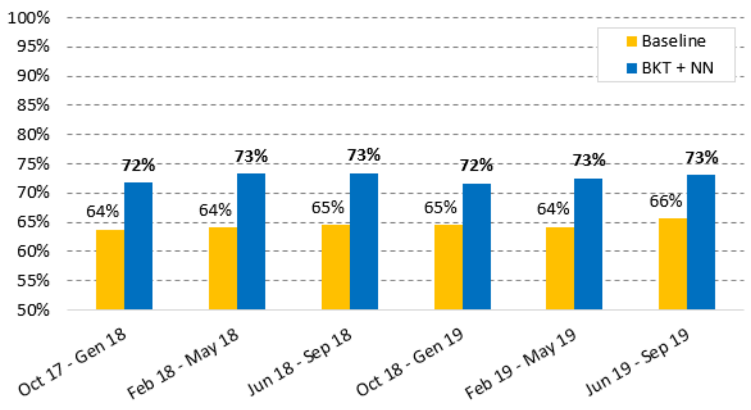


Figure 4.4: Baseline and BKTNN model performances

As you can notice in Figure 4.4, where the reported accuracy is truncated to the second decimal digit, both the methods provide almost constant results in the various tests based on the quadrimestri, proving their consistency.

Even if the simulated performances are not outstanding, we must take into account that:

- we developed the Knowledge Structure ourselves, inferring it from the platform's content and not using a pre-existing domain structure;
- we had no information about the users excluding the exam results;
- there is an elevated churn rate therefore a portion of students abandon the platform before the model, which has no initial information, has had time to effectively profile them;

- from the analysis of the exam results, we noticed an inconsistency in the students' answers: they frequently alternate correct answers to wrong answers, even if the subject and the subtopics required are the same. This aspect influenced the performances of our model since the BKT algorithm is constructed to quickly adapt to the students' answers.

Some options to boost the performances of the models could be:

- the creation of a more accurate Knowledge Structure, inferred by NLP algorithms and then fine-tuned by domain experts;
- a more accurate labelling of the concepts required to correctly answer the queries;
- the introduction of entry tests and surveys, to have prior knowledge of the students' skills.

Chapter 5

Recommender system

As we mentioned before, our goal consisted in the development a system capable of guiding each student along their optimal personalized learning path by recommending the next lesson to study. The system intervenes after a student has completed an exam, taking into consideration the student's knowledge of fundamental mathematical concepts and the underlying educational structure (see Figure 5.1).

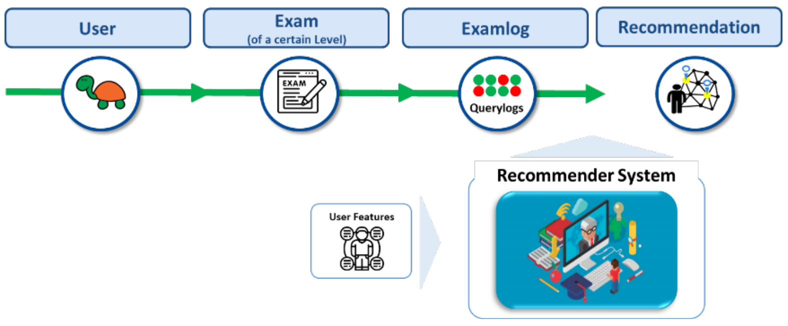


Figure 5.1: Recommendation process

The two previous steps served as the foundations to accomplish this goal. In fact:

- The creation of the Knowledge Structure allowed us to express the recommended learning path through the structure's concepts and relationships. In fact, the system recommends a lesson linked to the exam the student has just completed according to the Knowledge Structure's relationships;
- The user profiling process allowed us to create a model that estimates each users' knowledge of the different concepts and predicts future performances.

Recommender systems are algorithms aimed at suggesting relevant items (movies to watch, texts to read, products to buy, etc.) to users. In our case, the recommendation consists in a specific lesson that, if studied, should improve the students knowledge and guide them along the personal optimal learning path. These recommendations are made at the end of each exam, allowing the system to take into consideration the exam results and have a better comprehension of the actual user's knowledge and skills. Recommender systems can be classified into three main categories based on how the recommendation is obtained. The most popular ones are:

- Collaborative filtering (Figure 5.2) – recommendations are solely based on the past interactions recorded between users and items. The main idea behind collaborative methods is that the past user-item interactions are sufficient to detect similar users and/or similar items and make predictions based on these estimated similarities [**20**];

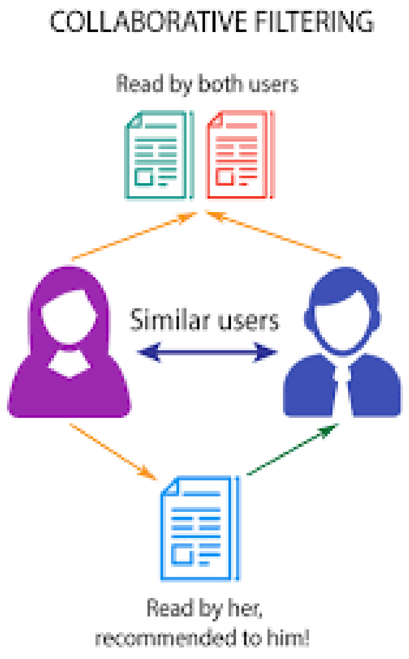


Figure 5.2: Example of recommendations based on the collaborative filtering approach

- Content based methods (Figure 5.3) – content-based approaches use additional information about users and/or items besides the past interactions. The idea behind content - based methods is to build a model based on the available "features" that explains the observed user-item interactions and reproduce these patterns in order to give the best recommendations [**21**];

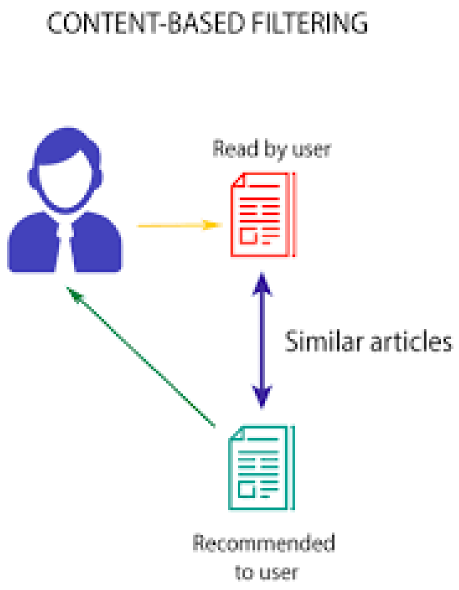


Figure 5.3: Example of recommendations based on the content based approach

- Rule-based recommendations - based on predefined rules that the developers of the algorithms impose according to domain expertise or analyses.

The lack of information about the users' preferences and the observation of the users' erratic behavior while navigating the platform's material led us to choosing a rule-based approach.

Based on the patterns observed in the historical data and the advice of Redooc domain experts, we developed a set of rules to guide the recommendation process. In particular, the recommendation corresponds to a certain level (see Section 1.1) and only in a second moment it will be associated to a lesson. This choice guarantees more precise instructions that also take into account the different difficulty levels of exams.

The implemented system is based on predefined rules and these rules derives from the analysis of the platform contents, of the users' performances and they have been elaborated with the assistance of educational expertise. Then, even considering its limitations, this system could be implicitly considered as an ontology-based recommender system [**22**].

5.1 Rule-Based Recommender System

Basically our system propose at the end of each exam 2 kind of recommendations. In fact, the recommendations are divided into default recommendations, based exclusively on the exam results and therefore independent of the student's level of knowledge, and adaptive recommendations, individualized per each student and situation. At the end of each exam, the system recommends one default recommendation and two adaptive recommendations.

The default recommendations are based on the users' history of failed and passed levels. In addition, they follow the general philosophy of "up or stay". In fact:

- If the percentage of correct answers is above 80%, the system recommends trying the first level which hasn't already been passed belonging to one of the following lessons, in accord to the lesson order we created (see Section 1.2);
- If the percentage of correct answers is between 60% and 80%, the system recommends trying the first level not already passed of the same lesson;
- If the percentage of correct answers is under 60%, i.e. the student has not passed the exam, the system recommends repeating the same level.

In this way, the student is forced to understand each lesson very well before being advised to proceed.

The adaptive recommendations, instead, are based on both the educational structure and the user history. The recommendations are always associated to levels that haven't been passed. Moreover, we define a lesson as "complete" if the student has passed all of its levels. The adaptive rules are as follows:

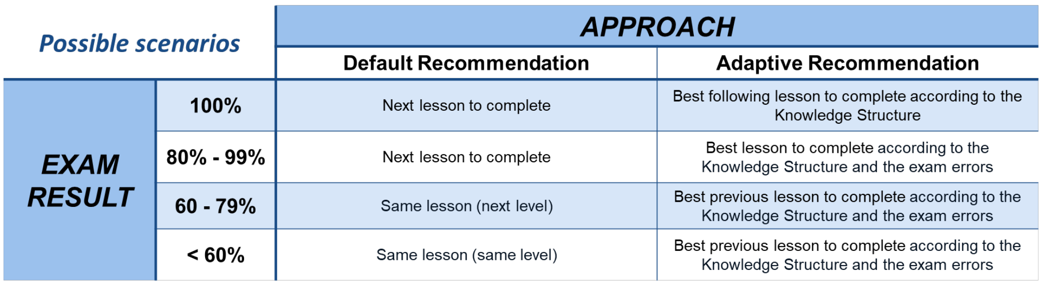
- If the student has correctly answered all the exam's queries, the system recommends trying the first non-passed level of the two following incomplete lessons;
- If the percentage of correct answers is above 80% and below 100%, the system recommends one level as in the previous case and one level belonging to previous lessons, determined based on the exam's errors. The system considers only the queries related to the wrong answers and collects their prerequisite lessons. These lessons are ranked according to the number of times they have been considered as a prerequisite for

the selected queries and according to their order on the platform (promoting the ones closest to the exam's corresponding lesson). Finally, the first incomplete level of the first incomplete lesson corresponds to the given recommendation. This method emphasizes the intention of recommending the best lesson to fill the gaps in the students' knowledge;

- If the percentage of correct answers is below 80%, the system recommends one level as in the previous case related to the wrong queries and one level according to the lessons prerequisite. The same procedure of the wrong queries is applied to select the lesson prerequisites. Finally, the first incomplete level of the closest lesson corresponds to the given recommendation. This method assumes a lack of general preparation for what concerns the lesson, where the knowledge gap is not related to the particular errors made during the exam but to the general knowledge of the subtopics present in the lesson. For this reason, we decide to emphasize the prerequisites of the lesson.

In case multiple exams related to the same level are carried out, the percentage to be considered is the last one obtained.

These recommendations rules can be summarized in Figure 5.4.



Possible scenarios		APPROACH	
		Default Recommendation	Adaptive Recommendation
EXAM RESULT	100%	Next lesson to complete	Best following lesson to complete according to the Knowledge Structure
	80% - 99%	Next lesson to complete	Best lesson to complete according to the Knowledge Structure and the exam errors
	60 - 79%	Same lesson (next level)	Best previous lesson to complete according to the Knowledge Structure and the exam errors
	< 60%	Same lesson (same level)	Best previous lesson to complete according to the Knowledge Structure and the exam errors

Figure 5.4: Recommendation schema

These recommendations have the goal of providing each user with a solid preparation that spans across all the concepts and lessons. In this way, the platform can help students fill knowledge gaps and get a broad education or guarantee a quick revision for those who simply want to review or elaborate upon lessons they have already seen.

This corresponds to the last version of the system implemented. In fact, time constraints forced us to simplify the rules in order to obtain a faster algorithm. The initial implementation provided adaptive recommendations based also on the predictions made by the knowledge tracing algorithms. More levels were considered and a simulation of the users results in the queries were performed. Then the considered levels were sorted on the base of the student's performances estimation favouring the closest one to the 70% percentage of correct answers. The expected results E_{exam} were calculated as starting from the following formula and dividing by the number of queries belonging to the exam:

$$\begin{aligned}E_{exam} &= \textstyle\sum_{query \in exam} E_{query} \\ &= \textstyle\sum_{query \in exam}[0 \cdot P(Action_{query} = Incorrect) \\ &\qquad +1 \cdot P(Action_{query} = Correct)] \\ &= \textstyle\sum_{query \in exam} P(Action_{query} = Correct)\end{aligned}$$

The percentage of 70% derived from the idea of recommending to the student lessons not too difficult to understand, but, at the same time, not so easy to result useless from the learning point of view. However, since almost the entire execution time of the system consists in executing time of the BKTNN algorithm, the first implemented version of Cassandra required an amount of time proportional to the number of simulation performed, making the response of too slow for a production environment.

5.2 Testing phase

During these activity years, Redooc had established collaborations with many different Italian schools. The latter have started to use the Redooc services in order to provide support to the students and to improve the general quality and level of teaching in the school. Thanks to this it has been possible to organize a future online test of the system with the student of the classes of one of these high schools.

The goal of the test was not only the evaluation of the system performances and behaviours in a production environment, but also the investigation of the educational impact it could have.

5.2.1 Test settings

The test would last 2 weeks, 4 or 5 hours per week, and would involve 10 classes, 2 per scholastic year, of a "Liceo Scientifico" ("Scientific High School"). Each class would be divided into 2 homogeneous groups, group A and B (Figure 5.5), in terms of skills and knowledge according to their math teacher indications. Then, during the first week group A would not receive recommendations while Group B would be forced to use the Recommender

System, and during the second week group A would be forced to use the Recommender System while B would not receive recommendations. Moreover, one class per academic years would begin a lesson related to a topic that the students are studying in class during the period prior to the starting date of the test, while the other would begin with a new topic.

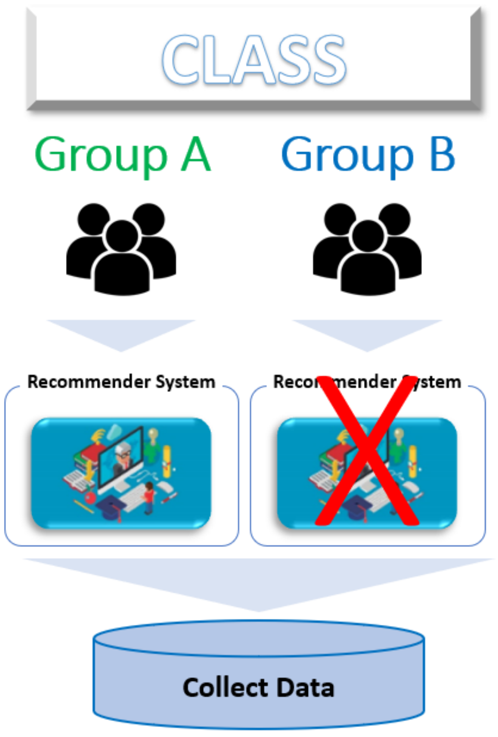


Figure 5.5: Test schema

5.2.2 Data Collection

During the entire test it would be possible to monitor the users responses and the analyze the results of the tests.

In addition to that, we have created 3 exams about general math questions to test the students before the first week, between week 1 and week 2 and at the end of the second week. These tests should be able to give us an overview about the level of preparation of the students before, in the middle and after the test, as well as its changes using or not using the recommender system.

Finally, a questionnaire would be proposed to the students in order to collect their feedbacks about for example the utility, the efficiency, the performances of the Adaptive e-Learning System used. In fact, when developing an ALS, it is important also to take into consideration psychological and social aspects like its impact, the general level of acceptance, the ease of use for the students, the psychological and social benefits and/or drawbacks.

Chapter 6

Conclusion

In recent years, we have seen the spread of Machine Learning approaches and algorithms in several different fields, with particular attention to Neural Networks and their ability of solving a great variety of problems. In our case, alongside a proper analysis, they have successfully performed human tasks like the structuring of educational material and the understanding of the student knowledge.

Starting from the educational material, that correspond to a set of unstructured data sources, we were able to extract the main underlying concepts through the usage of topic modelling techniques and we were able to express each document as a combination of these concepts. Then, combining this representation of the data with the one based on Word Embedding techniques we have found the relationships among the documents and we have created a representation of the Knowledge Structure related to the Mathematical program of study of the Italian High School. The Knowledge Structure represents the core of an Adaptive e-Learning System since it is the base for both the creation of the platform structure and the modelling of the users. In our case the platform have been already developed, so we concentrate only on the minimal navigation changes needed to guarantee the presentation of the recommendations. This choice allows to be not invasive in a pre-existent system.

With the Knowledge Structure it was possible to model the students and to trace their level of mastery per each base concept. In this phase we have investigated different solutions that take into account also the possibility of problems involving multiple concepts. We have combined the traditional Bayesian Knowledge Tracing algorithm with the Neural Networks to be able to overcome the limitations imposed by the presence of correlation among the topics and the way the users have learnt them. In order to do this it was necessary to consider all the available information about the student that we were able to extract from their historical data on the platform.

These general lacks of information about the users have forced us to the implementation of a rule-based recommender system based on the Knowledge Structure. A collection of data more complete could certainly improve the efficiency and the potential of the Adaptive e-Learning System, also allowing the consideration of more behavioural aspects like the user's preference and learning style.

Finally, we have enclosed all these approaches and algorithms in a unique system named "Cassandra". We were able to elaborate a procedure that starts from the educational contents and the exams results, extracts a Knowledge Structure and finally models the users. Alongside this procedure, we were able to implement from end to end a simple and fast Adaptive e-Learning System capable of analyzing the users' performances in a production environment, trace their knowledge and propose the best educational contents to guarantee them a solid preparation in each concept.

Cassandra represents a valid support both for the students and the teachers that, with this system, will be able to understand at any moment the level of knowledge of their scholars and analyze the personalized recommendations proposed. Then, they could elaborate lessons and assign homework taking into consideration these information. These are just the most obvious applications and benefits of such a system, but it and the procedure followed in its implementation could be used in many different fields and with different scopes. In fact, the Adaptive e-Learning Systems represent the frontier of the e-Learning and will in the future play a central role in the education sector.

Bibliography

[1] C. Romero, S. Ventura, *Educational Data Science in Massive Open Online Courses*, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2017

[2] P. Brusilovsky, *Methods and Techniques of Adaptive Hypermedia*, 2017 Eighth International Conference on Intelligent Computing and Information Systems (ICICIS), 2017

[3] D. Onah, J. Sinclair, *Massive Open Online Courses – An Adaptive Learning Framework*, Conference: In Proceeding of the 9th International Technology, Education and Development, 2015

[4] P. Brusilovsky, *Developing Adaptive Educational Hypermedia Systems: From Design Models to Authoring Tools*, 2003

[5] D. Blei, A. Y. Ng, M. I. Jordan, *Latent Dirichlet Allocation*, Journal of Machine Learning Research 3, 2003

[6] van der Maaten, L.J.P.; Hinton, G.E., *Visualizing High-Dimensional Data Using t-SNE.*, Journal of Machine Learning Research 9:2579-2605, 2008.

[7] Piotr Bojanowski, Edouard Grave, Armand Joulin, Tomas Mikolov, *Enriching Word Vectors with Subword Information*, 2016.

[8] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, Luke Zettlemoyer, *Deep contextualized word representations*, 2018.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*, 2018.

[10] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean, *Efficient Estimation of Word Representations in Vector Space*, Proceedings of Workshop at ICLR, 2013.

[11] Quoc V. Le, Tomas Mikolov, *Distributed Representations of Sentences and Documents*, 2014.

[12] Sein Minn, Yi Yu, Michel Desmarais, Feida Zhu, Jill Jenn Vie, *Deep Knowledge Tracing and Dynamic Student Classification for Knowledge Tracing*, 2018.

[13] M. Desmarais, R. Baker, *A Review of Recent Advances in Learner and Skill Modeling in Intelligent Learning Environments*, 2012.

[14] A. Corbett, J. R. Anderson, *Knowledge tracing: Modeling the acquisition of procedural knowledge*, User Modeling and User-Adapted Interaction, 1995.

[15] J. Gonzalez-Brenes, Y. Huang, and P. Brusilovsky, *General features in knowledge tracing to model multiple subskills, temporal item response theory, and expert knowledge*, The 7th International Conference on Educational Data Mining, 2014.

[16] Phil Pavlik Jr, Hao Cen, Kenneth R. Koedinger, *Performance Factors Analysis - A New Alternative to Knowledge Tracing*, Conference: Artificial Intelligence in Education: Building Learning Systems that Care: From Knowledge Representation to Affective Modelling, Proceedings of the 14th International Conference on Artificial Intelligence in Education, 2009.

[17] Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas Guibas, Jascha Sohl-Dickstein, *Deep Knowledge Tracing*, Advances in Neural Information Processing Systems 28, 2015.

[18] R. Baker, A. T. Corbett, V. Aleven, *More Accurate Student Modeling Through Contextual Estimation of Slip and Guess Probabilities in Bayesian Knowledge Tracing*,Proceedings of the 9th International Conference on Intelligent Tutoring Systems, 2008.

[19] Diederik P. Kingma, Jimmy Ba, *Adam: A Method for Stochastic Optimization*,3rd International Conference for Learning Representations, 2015.

[20] Yue Shi, Martha Larson, Alan Hanjalic, *Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges*, ACM Computing Surveys (CSUR), 2014.

[21] Robin Van Meteren, Maarten Van Someren, *Using Content-Based Filtering for Recommendation*, Proceedings of the Machine Learning in the New Information Age: MLnet/ECML2000 Workshop, 2000.

[22] John K Tarus, Zhendong Niu, Ghulam Mustafa, *Knowledge-based recommendation: a review of ontology-based recommender systems for e-learning*, Artificial intelligence review, 2018.

[23] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov, *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*, Journal of Machine Learning Research, 2014.

[24] Tong Zhang, Bin Yu, *Boosting with early stopping: Convergence and consistency*, The Annals of Statistics, 2005.

[25] *Massive open online course*, Wikipedia.