X Modal
X Cultural
X Lingual
X Domain
X Site
Global OER Network

Grant Agreement Number:	761758
Project Acronym:	X5GON
Project title:	Cross Modal, Cross Cultural, Cross Lingual, Cross Domain, and Cross Site Global OER Network
Project Date:	2017-09-01 to 2020-08-31
Project Duration:	36 months
Deliverable Title:	D3.2 – Learning Analytics
Lead beneficiary:	Nantes
Type:	Report
Dissemination level:	Public
Due Date (in months):	24 (August 2019)
Date:	
Status (Draft/Final):	Final
Contact persons:	Colin de la Higuera, Walid Ben Romdhane and Victor Connes

Revision

Date	Lead author(s)	Commments
17-Aug-2019	Alfons Juan	Formal review of the deliverable
31-Aug-2019	Colin de la Higuera	Final version after Alfons' review

Contents

List of Figures

List of Tables

Abstract

Learning analytics allow us to extract more information from the content and user data collected by X5GON. This extra information serves principally two purposes: (1) to be used by other X5GON components to provide high level returns such as recommendations, learning activities, learning paths, and (2) to help the X5GON developers and researchers imagine new ideas and opportunities based on a better perception of what the data can tell us.

The Learning Analytics Machine 2 (LAM) is built upon the data collected by X5GON in years 1 and 2 and the LAM 1 delivered in 2018.

The LAM consists of (1) an Application programming interface (API) with access of many different models and (2) a dashboard allowing to view several analytics.

The API and the dashboard can be accessed online : http://wp3.x5gon.org.

We also present a number of separate analyses and preliminary experiments which will result in further models for year 3 of the project.

1 Introduction

Learning analytics allow us to extract more information from the content and user data collected by X5GON. This extra information serves principally two purposes: (1) to be used by other X5GON components to provide high level returns such as recommendations, learning activities, learning paths, and (2) to help the X5GON developers and researchers imagine new ideas and opportunities based on a better perception of what the data can tell us.

1.1 What is the Learning Analytics Machine?

The Learning Analytics Engine aims to allow the analysis of the learning and testing aspects of X5GON, and links with educational theories, affective computing, etc. including the cross-modal, cross-lingual and cross-cultural aspects.

The LAM 2.0 is essentially an API allowing access to information about the data collected and thus both about the learning material and the user activities. This information can be visualized through a dashboard which also allows a simple navigation through the accessible learning material. The dashboard can accessed here: http://wp3.x5gon.org.

1.2 About the LAM 1.0 delivered in 2018

In D3.1 was presented the LAM 1.0 as well as different side analyses concerning users and content. The API allowed the access to two models. We also presented some early experiments with the user intent, the navigation paths and the capacity of finding a missing lecture.

During the 2018 Y1 review report these were the comments and remarks which have the strongest impact on the work done:

- In "General comments": *In some cases, it is difficult to identify innovative parts and novel contributions of the project. For example, the techniques for text similarity and user modeling in WP3 and WP4 do not demonstrate going beyond the existing SoA.*
- In "Recommendations concerning future work, if applicable": *In addition, it will be important to clearly demonstrate the innovative contributions of the project as defined in GA in the proposal in the following period, in particular concerning user modelling techniques, OER retrieval methods, machine learning, etc.*
- In "Annex 1 - Expert's opinion on deliverables", on D3.1 – Learning Analytic Engine 1.0: *Future versions should emphasize on novel contribution in relation to the SoA, deeper analytics (AI and ML methods), user evaluation of techniques and especially the graph- based GUI.*

In the rest of the report we have aimed to respond to these issues. Several innovative elements have been introduced in Year 2. To name a few:

- The content models which have been built and are described in Section 2.1.2, 2.1.7 are new. They allow to describe the change of dominating concepts through time in a resource and can then be used either to better identify concepts linking chronologically 2 resources (Sec. 2.2.2) or to extract a fragment of resource for recommendation instead of an entire resource.
- The user models described in Section 3.1-3.3 are new. They allow to associate a most probable resource path to a user, given only a high level description of his actions. The section is mathematical and still requires an implementation.
- The graphical representation (Sec. 4.2.1) of a resource in its environment (closely linked resources with indications of length and relative hardness) is new.
- The recommendation system described in Section 3.4 is based on a Probabilistic Bayesian Network, with an architecture specifically designed for the project.

The choices made have not led to presentations outside working groups. This will be where efforts will be needed in year 3.

1.3 Main achievements

In this document we present the work done in WP3 around the leaning analytics machine.

Several aspects are presented, including:

1. The API itself which gives access to more than 10 models. Some of these are called *intermediate* (Section 2.1) as they are obtained through a direct analysis of the data. Others are *rich* (Section 2.2) in the sense that they are built from a use of (often new) algorithms on this data, for an enhanced quality of information (Section 4.1).
2. A description of the graphical and visual representation of some of the API models (Section 4.2).
3. A first set of tools and methods allowing to evaluate the complexity of a resource (Section 2.2.1).
4. Experiments showing how to obtain a missing resource, a next resource, a previous resource (Section 2.2.4).
5. A set of methods and techniques allowing to visit a user path and to reconstruct the set of resources the user has consumed. From these, a topic vector corresponding to a user is built (Section 3).
6. A Probabilistic Relational Model used to make recommendations (Section 3.4).
7. The Moodle plug-in which allows to get hold of the content data from Moodle platforms, Moodle being the most popular Learning Management System (Section 5.3).

2 Content Analytics

Content has been recovered from many different repositories, through crawling and the use of ConnectService. This content is transcribed (if and when necessary) and then translated. We therefore work from text.

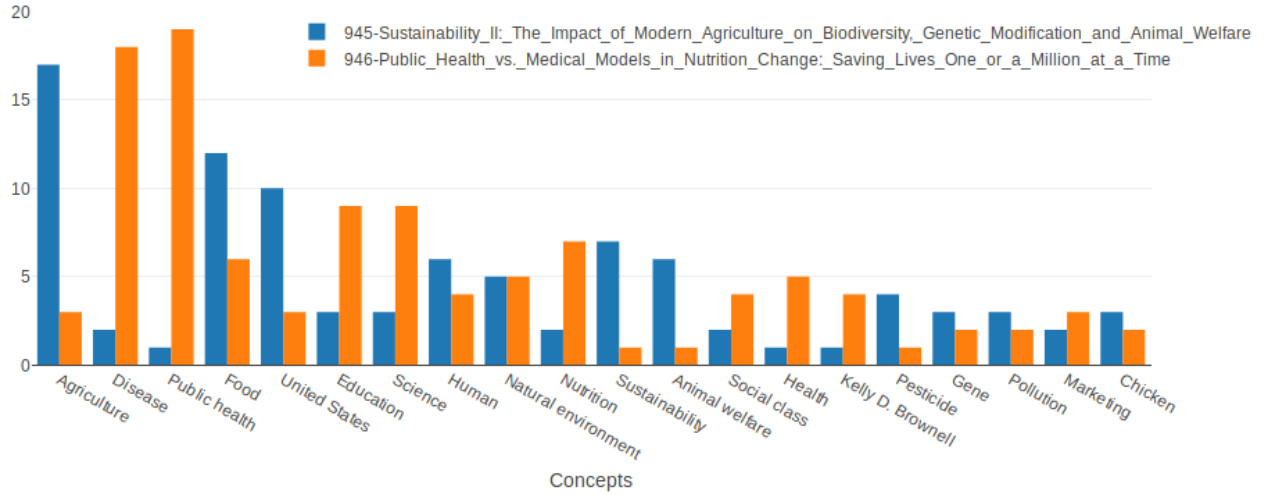

Figure 1: Example of comparisons of concepts extraction for two resources

The goal of *Content Analytics* is to capture essential features about this data: what is a course about? how do 2 courses relate? Questions also concern users: what does this user know? What is he interested in? How close are these users?

The API allows access to all these models.

2.1 Intermediate models

In this section, we describe all intermediate models available in the API. For each, a call of the corresponding service is provided as an example.

2.1.1 Wikifier

This model represents a text as a distribution of Wikipedia concepts [1]; each Wikipedia page represents a particular concept. This is represented in Figure 1 where two similar resources are viewed through their (common) Wikifier concepts. This model has the advantage to enable cross-lingual comparisons of documents. The API provides two services linked with this model. The first is called *getresourcewikifier* and allows to recover the concept extraction of document, ie. the vector of concept scores for the document. The second one is called *wikification* and allows, for a given document, to recover its k nearest neighbours in the embedded WIKIFIER space. Details of their usage can be found in Section 4.1.2, and a discussion about the pros and cons of the WIKIFIER is also available in Appendix 8.2.

2.1.2 Continuous Wikifier

One of the main weaknesses of the WIKIFIER is to lose the temporal information about concepts; indeed, each document is represented as a single concept extraction (vector). One way to capture the evolution of concepts over the length of the document is to compute the WIKIFIER for each chunk of 5000 words.

In this way, the text is no longer represented by a vector over the Wikipedia concepts but is now represented by a matrix where each line represents the concepts extracted for a chunk of the input text. If we consider an overlapping between chunks, we obtain a continuous representation of the

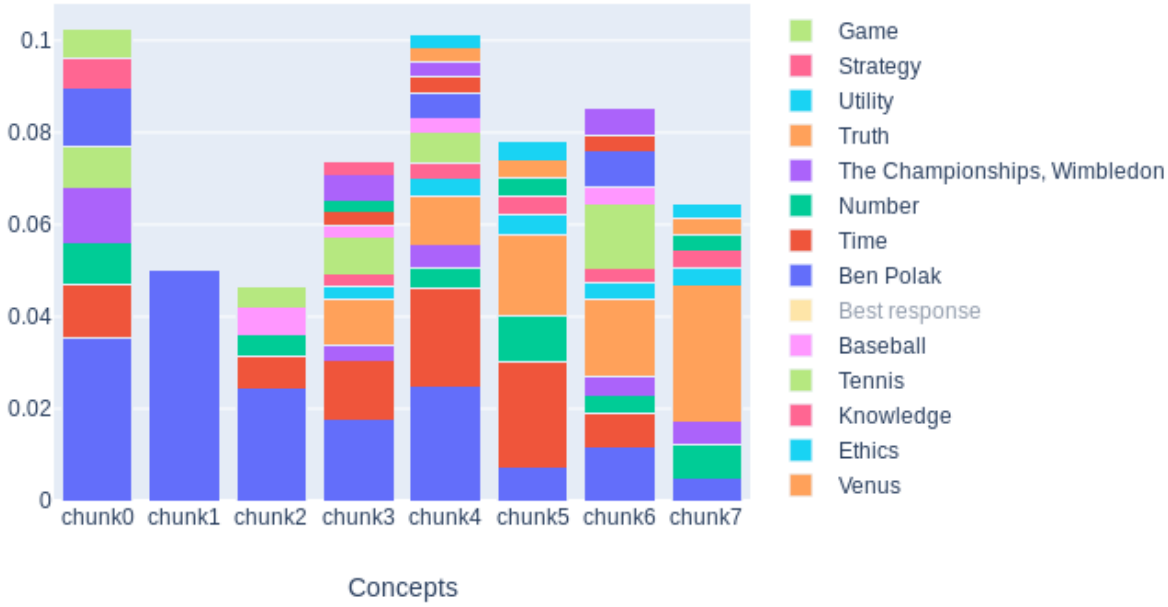

Figure 2: Follow evolution of distribution over concepts in a document using continuous Wikifer

concepts through the input text. As shown in Figure 2 we can thus follow and compare distribution over concepts for each chunk in the text.

2.1.3 Tf-Idf

Tf-Idf is one of the more traditional way of representing a text in a corpus; this method is based on Salton's work [2, 3, 4]. A text is represented as a distribution over the corpus vocabulary, the value for a given term and a given text increases when the term is frequent in this document and infrequent in the rest of the corpus. The services we provide allow to recover the Tf-Idf (learnt on the whole database) for a given text and to compute the k-nearest neighbours of a text in the Tf-Idf space.

2.1.4 Processed text

The API also provides 3 basic tools for text preprocessing:

- One service to remove stop words (English only);
- One service for text lemmatization;
- One service which removes stop words, lemmatizes, and keeps only nouns, proper nouns, verbs and adjectives.

2.1.5 Phraser

The phraser is another text preprocessing introduced by Quoc V. Le and Tomas Mikolov [5], which is particularly useful before creating word embeddings. The aim of this method is to capture phrases

that have a meaning that is not a simple composition of the meanings of their individual words, which is particularly the case for named entities (eg: New York, Toronto Raptors, Larry Page...) and to represent them as a single token. In order to do that, the method automatically detects words which appear frequently together, and infrequently in other contexts.

2.1.6 Doc2Vec

In the Doc2Vec [6] model each document is represented by a dense vector; the learning of these vectors is done using encoder approaches. These vectors will be close for the cosine distance in the embedding space when the two corresponding document are semantically close.

In the first approach, a 1-layer neural network is used to predict a central word for a given context (words which surround the central word in a sentence) and a `document_id`. This method is called *Distributed Memory version of Paragraph Vector* (PV-DM).

The second approach is based on an opposite philosophy where the aim is to predict a sentence (context + central word) given only the `document_id`. This method is called *Distributed Bag of Words version of Paragraph Vector* (PV-DBOW).

In both cases, the vector representing each document is directly recovered from the encoder. In the article, the authors state that they recommend using a combination of both algorithms, though the PV-DM model is superior and usually will achieve state of the art results by itself.

2.1.7 Continuous Doc2Vec

The Continuous Doc2Vecfollows the same idea as the continuous Wikifier: instead of keeping only one vector by document, we cut the document into regular fixed size chunks, and learn a vector for each of these chunks. Empirically, we observe that vectors for chunks of the same document are close in the embedding space (see Figure 8).

2.2 Rich models

Recommendation in the context of educational resources raises questions that are quite different from those in a commercial context. Indeed, our goal is not to maximize the user's conversion rate, or watching time, but to improve the learner's learning experience. We conjecture that this aim is closely connected with the intent of the learner. One learner may search for an easier resource to explain a concept. Another learner may want to dig deeper into a concept. And yet another learner may fix herself a goal and require a learning path between her knowledge and this goal...

To answer such questions it seems necessary to define a background ontology between the resources. In this section, in order to begin to build this underlying ontology, we define tasks that may be parts of it, models to solve these tasks and try to evaluate them on human annotated corpus.

2.2.1 Complexity

One essential feature of a course is that it is intended by design for an audience. Yet in very few cases does the meta-data tell us who the target is. We have started working on being able to predict this target audience from content. On the long run, user information may also be of help. But there will always be a *cold start* issue here, for example with new lectures. As a first task, we have attempted to answer the following question: "What age group is this course intended for?"

Understanding the difficulty/complexity of a task or course can be done in various ways: *heuristically*, using estimates provided by experts or *mathematically*, considering a set of characteristics, for example the number of concepts involved. This distinction is in line with the difference that a community of authors [7, 8] makes between the concept of difficulty and complexity: while *difficulty* relates

to relative complexity, *complexity* can be defined in a more objective way through text properties such as lexicon and grammar, as we did here.

Detecting age groups. Complexity of a text can be approached through the prism of *readability*. More precisely, *"a reading difficulty, or readability, measure can be described as a function or model that maps a text to a numerical value corresponding to a difficulty or grade level"* [9]. Inputs are usually statistics related to lexical and/or grammatical features of the text and output is generally a difficulty grade level. The level of a resource and the age of its target audience are intrinsically linked, as the Table 1 reminds us.

Educational stage	Grade	Ages
Preschool/Kindergarten	Preschool/Pre-K	4-5
	Kindergarten/K	5-6
Elementary School	Grade 1	6-7
	Grade 2	7-8
	Grade 3	8-9
	Grade 4	9-10
	Grade 5	10-11
Middle School	Grade 6	11-12
	Grade 7	12-13
	Grade 8	13-14
High School	Grade 9	14-15
	Grade 10	15-16
	Grade 11	16-17
	Grade 12	17-18

Table 1: Ages in grades (US system)

Although there is no list of text complexity parameters nor any range of parameters playing a role on the text complexity, assessing the complexity of a course can be done through basic quantitative parameters. Many complexity formulas have been proposed on such a basis [10, 11, 12, 13]. Among the parameters we have decided to reuse here, we can mention: the sentence and word length, the number of long or difficult words, the number of distinct words and the number of specific lexical items (adverbs, pronouns). The Flesch-Kincaid [14] *reading ease* formula based on the above-mentioned parameters was also used, as well as two complexity metrics defined internally: the Kurtosis criteria and the number of concepts over time which are explained bellow (see Section 2.2).

Our task - predicting an appropriate grade for a given resource - was modelled as a regression problem. A first step related to the collection of annotated data was required beforehand, as mentioned in Section 5.1. The courses retrieved were aimed at students aged between 4 and 18 years old, and covered many subjects such as Mathematics, Life Sciences, English Language Arts, History and Social Sciences. The distribution of resources by grade is shown in Figure 3.

Each of the above-mentioned criteria was individually assessed (cf. Table 2). Results tend to show that the most discriminant features for estimating the grade of a resource are the number of words per sentence, the Flesch reading and grade level tools and the number of difficult words in a text. In other words, the longer are the sentences and the more complicated the vocabulary used is, the more complex the resource will be considered. On the other hand, part of speech tags such as adverbs, pronouns and conjunctions do not seem to really guide the assignment of a level to a resource since the estimated resource level is always done to within two grades. We can also note that the distribution

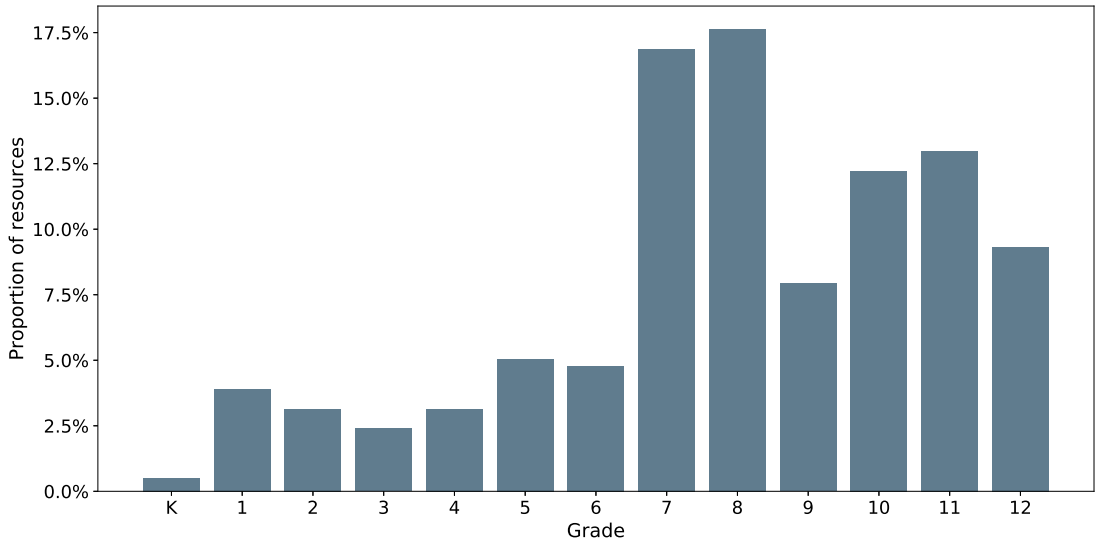

Figure 3: Grade distribution

of concepts processed over time is neither significant for the level of complexity of a resource.

Since there did not appear to be any linear correlations between the features (except between the Flesch measures), we decided to keep them all in our models. Two models were tested: first a regression tree to predict the exact grade of a resource and then a gradient boosting in order to predict the range of grades for which a resource was designed since some of them covered several grades. These two models were chosen for their ease of interpretation, but also because they make it easier to select features.

A regression tree was first trained on the data collected on the *Amazon Inspire* (Section 5.1) platform and others open educational resource websites such as CK12[2] (Section 5.1). As it appears in the results, the number of words per sentence has separated the resources into two main groups: the first one covering grades K to 8 and the second one grouping the resources dedicated to high school. The number of difficult words and the number of syllables per word then appear as the most discriminating criteria.

To generate prediction intervals instead of a single grade for each of the resources, three gradients boosting were then trained. Results are provided in Table 3. The results appear to be better than with the first model, especially for the mean value. The prediction made by the model is correct to within 1 grade (cf. MSE values).

Even if the features evaluated may seem superficial to describe the complexity of a text, the results of both evaluations are good.

New metrics. Based on features extracted from the API, we defined new metrics in order to approximate the complexity. The two metrics below assume complexity can be inferred by studying the distribution of concepts over the document.

[1]https://github.com/mauryquijada/word-complexity-predictor/blob/master/README.md
[2]https://www.ck12.org/student/

Feature evaluation			
Feature	Description	Training score	Testing score
word_sentence	number of words per sentence	1.68	1.72
syllab_word	number of syllables per word	1.94	1.86
distinct_word	number of distinct words in the text	2.02	2.18
concreteness	the concrete nature of a word (as opposed to abstract)[1]	2.09	2.09
flesch_reading	assesses the readability of texts	1.75	1.76
flesch_grade	provides a score as a U.S. grade level	1.75	1.78
difficult_words	number of difficult words	1.81	1.87
nb_adverbs	number of adverbs in the text	2.06	2.12
nb_pronouns	number of pronouns in the text	2.05	2.37
nb_coor_conj	number of conjunction, coordinating	2.14	2.13
character_word	number of characters per word	1.91	1.90
kurtosis	*See below*	2.25	2.27
concepts_sec	number of concepts per second	2.17	2.17

Table 2: Evaluation of the features of the regression model through the Mean Square Error (MSE)

Models	Predictor	Training score	Testing score
Gradient boosting	mse_min	0.63	1.13
	mse_mean	0.66	1.13
	mse_max	0.71	1.20
Regression tree	mse_mean	1.06	1.61

Table 3: Evaluation of the features of the boosting regression model through the Mean Square Error (MSE)

Population	Reading speed
Median Kindergarten graduate	10 wpm
Median Grade 1	23 wpm
Median Grade 2	72 wpm
Median Grade 3	92 wpm
Median Grade 4	112 wpm
Median Grade 5	140 wpm
Median Grade 8	250 wpm
Median university student	450 wpm
Median mid-level executive	575 wpm
Median university professor	675 wpm
World champion	4.700 wpm
Median adult	**300 wpm**

Table 4: Reading speed in the population (in word per minutes). Source [15]

Concepts over time. The concepts over time is the most straightforward of these approaches; the hypothesis behind is quite simple: The more a resource addresses concepts, the more this resource is complex. More intuitively, debate between experts will need many notions to be understood when a tutorial on the addition need very few requirements.

In order to obtain a quantitative evaluation for it, we simply choose to count the number of WIKIFIER concepts referenced by character in the document. So, $concepts\_sec(d) = |Wikifier(d)|/len(d)$, we decide to express this value in a more understandable way, using the average speed reading of a human being, finally the metric is expressed in $concepts/seconds$. This unit of measurement is particularly relevant in the task presented above (predicting the appropriate age for a resource), because the reading speed is strongly correlated with age during the period of childhood [15]. Table 4 shows examples of reading speed over the population. Note that, for the explanation we choose WIKIFIER as concepts extractor but the method is merely generalizable for all methods of concepts extractions.

Kurtosis. One of the big weaknesses of concepts over time is the lack of consideration for the importance of the concepts in the document. Indeed, we can easily imagine a scientific mediation resource which addresses many concepts very shallowly, and remain easy to access (case 1). At the opposite, a debate between philosophers can also be focused on a very specific set of complex points, such that, few concepts are addressed but very deeply, leading to a complex resource (case 2). The Kurtosis is a measure of the *tailedness* of the probability distribution. And seems to be, a pertinent indicator to the complexity regarding the previous examples. In particular, if we follow the Moors interpretation [16], high values of kurtosis arise in two circumstances:

- where the probability mass is concentrated in the tails of the distribution (case 1).
- where the probability mass is concentrated around the mean and the data-generating process produces occasional values far from the mean (case 2).

Formally, the Kurtosis of a distribution is defined as

$$Kurtosis(Wikifier(d)) = \mathbb{E}\left[\frac{Wikifier(d) - \mu}{\sigma}\right]$$

.

2.2.2 Ordering resources

Let's assume the existence of a series S of N document $S_1, ..., S_i, ..., S_N$ defined by humans. We want to learn a function f such that for each pair of document $\{S_i, S_{i+1}\}$, f is able to predict whether S_i has to be consulted before S_{i+1} or after.

One way to do that is to follow the evolution of concepts through the document; intuitively the document S_i should define concepts which can be reused in the document S_{i+1}. More precisely, the common concepts should be introduced earlier than in S_{i+1}, due to the fact that the learner is familiar with them since they have already been mentioned in S_i.

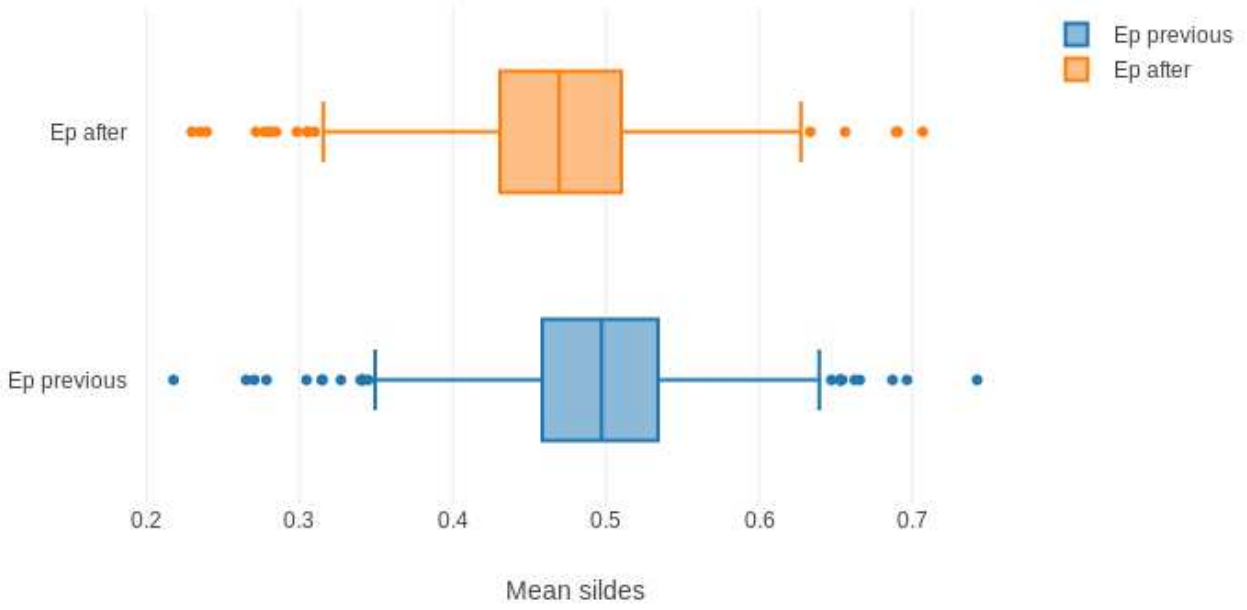

Figure 4: Average chunk of apparition of common concepts for S_i and S_{i+1}

Using the Continuous Wikifier (see Section 2.1.2) and following the above assumption, we can find the right order for $> 62\%$ of document pairs in the YaleOpenCoursewareCorpus (see Section 5.1).

In Figure 4 we show where, in average, the common concepts appear for S_i and S_{i+1}.

By deepening the analysis we contend that the particularly interesting concepts are those which are present in the end of S_i and in the beginning of S_{i+1} as positive examples and those which are present in the end of S_{i+1} and in the beginning of S_i as negative examples. Figure 5 shows the evolution of main common concepts between two resources (both courses about *Politics of Food*) the resources have been well ordered for the visualization.

In this example the four concepts validate our hypothesis and no negative samples are observed. In order to take benefit of these hypotheses, we chose to create a vector of evolution for each common concept c in our two resources (S_i, S_{i+1}), respectively called v_{c,S_i} and $v_{c,S_{i+1}}$. Vectors for the four concepts of Figure 5 are shown in Figure 6.

It suffices now to multiply these two vectors $M_c = v_{c,S_i} v^T_{c,S_{i+1}}$ to obtain a predicted order based on each concept. By considering the top right value $M_{c_{\{end,beg\}}}$ and the bottom left value $M_{c_{\{beg,end\}}}$ of the obtained matrix M_c, we calculate $pred_c = M_{c_{\{beg,end\}}}$ - $M_{c_{\{end,beg\}}}$ when $pred_c < 0$, c is a negative example and leads us to predict the order (S_{i+1}, S_i), whereas when $pred_c > 0$, c is a positive example and lead us to predict the order (S_i, S_{i+1}).

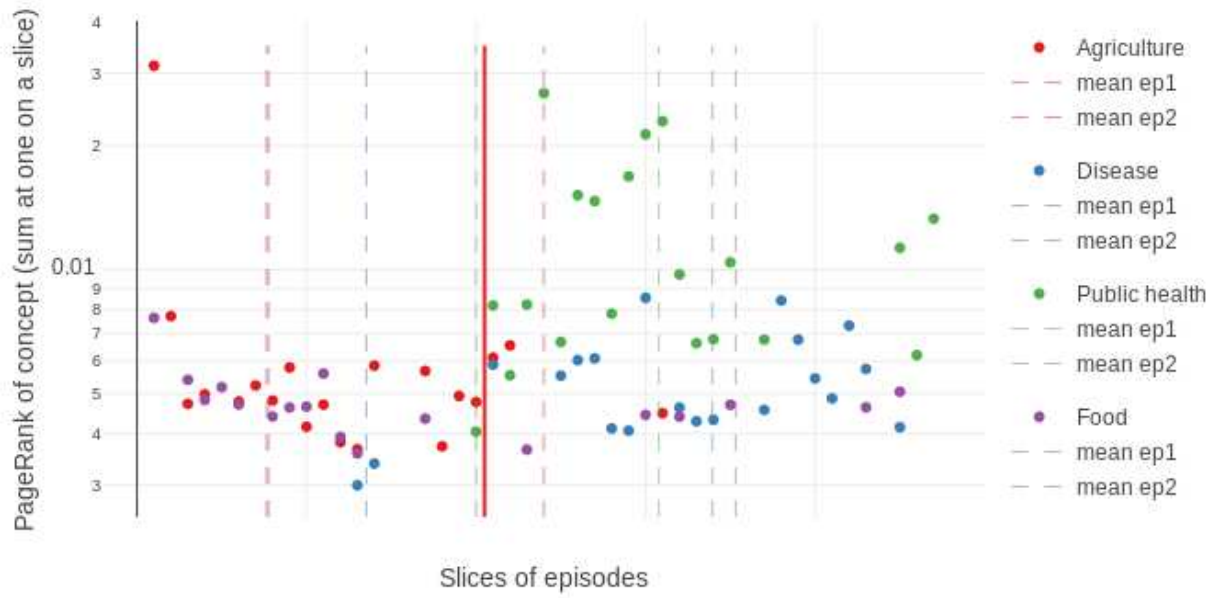

Figure 5: Evolution of main common concepts between two resources about "Politics of Food"

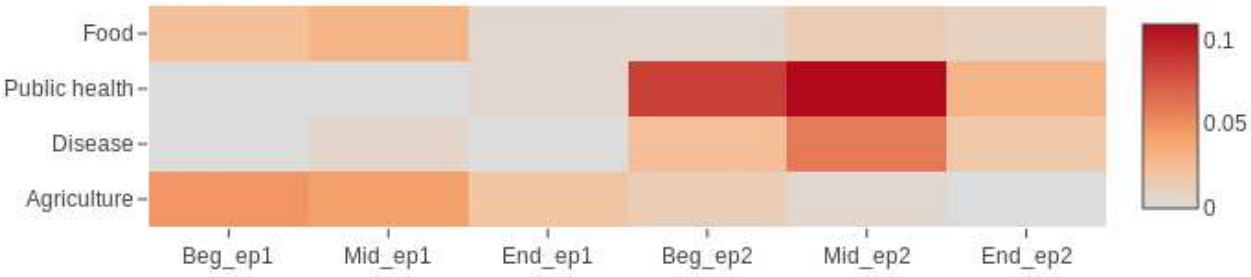

Figure 6: A user's transaction in origin database

We can now aggregate these predictions in order to obtain the final one. We choose to also apply a normalization term in order to reduce the impacts of artifact concepts returned by the WIKIFIER (see Appendix 8.2). Especially, our approach tries to catch basic concepts of the presentation and not specific ones which are present only very few times in the lecture; this is for example the case for named entities. In order to counter this effect we weight each concept's prediction by the inverse of the variance of the importance of the concept at each slice, more formally, let us define $pred_{S_i,S_{i+1}}$ as the final prediction,

$$pred_{S_i,S_{i+1}} = \mathbb{E}_{\forall c \in Common(S_i,S_{i+1})}\left[\frac{pred_c}{\sigma^2([v_{c,S_i}|v_{c,S_{i+1}}])}\right]$$

Finally, we can predict the order regarding the sign of $pred_{S_i,S_{i+1}}$: for a positive value we predict S_i then S_{i+1}, and the opposite order for a negative one. With this method we obtain $> 84\%$ of well ordered document pairs in the YaleOpenCoursewareCorpus.

2.2.3 Next resource prediction

In this section, we present a work in progress on content recommendation, only from content analysis.

Principle. Being able to recommend a resource adapted to the user is an essential question. In fact, it is the key question of X5GON. In this section, we focus on predicting the resource that should be consulted by the user, but only through information about the content of the resource. The use of other user-type information will later be incorporated into the model. In order to produce an estimate of this following resource, we consider a course as a sequence of sessions and a session as a sequence of chapters.

First, the documents are projected into a new representation space using DOC2VEC [6]. This new representation subsequently allows documents to be considered using neural models. Given that the sessions are seen as sequences, we use an LSTM model, specifically adapted to this type of data –see Figure 7– to predict the following chapter. Working on chapter scale, each chapter is represented by its own DOC2VECvector and a sequence by a matrix produce by the concatenation of DOC2VECvector of each chapter in the sequence. In this experiments the scope of the sequence is limited at the session or at the course depending on the grouping parameter.

Various insights. All model were trained when required and evaluated on YaleOpenCourseware; we operates a 20/80 split between the testing set and training set.

Inputs. In order to be input in the LSTM architecture chapter transcriptions must be passed as numerical vectors. One of the most common technique to transform string documents into vectors is DOC2VEC. Implemented as a Python library in Gensim [17], it allows unsupervised training on our documents in order to represent them as feature vectors.

Features should represent semantic proximity between documents. A way to evaluate the quality of our inputs is to show the similarity matrix between our chapters. In Figure 8, we can observe that chapters from the same courses are forming clusters. We can deduce that chapters with similar vocabulary are close one to each other. This observation is confirmed by the PCA three dimensional representation of the same embeddings (Next-Resource project).

A similar clustering was performed with one vector for each session and a clustering over the course.

In the LSTM the size of input context is one of the central questions; in our case we chose to give as input all the previous chapter in order to predict the following one. Alternative choices could be considered in the future.

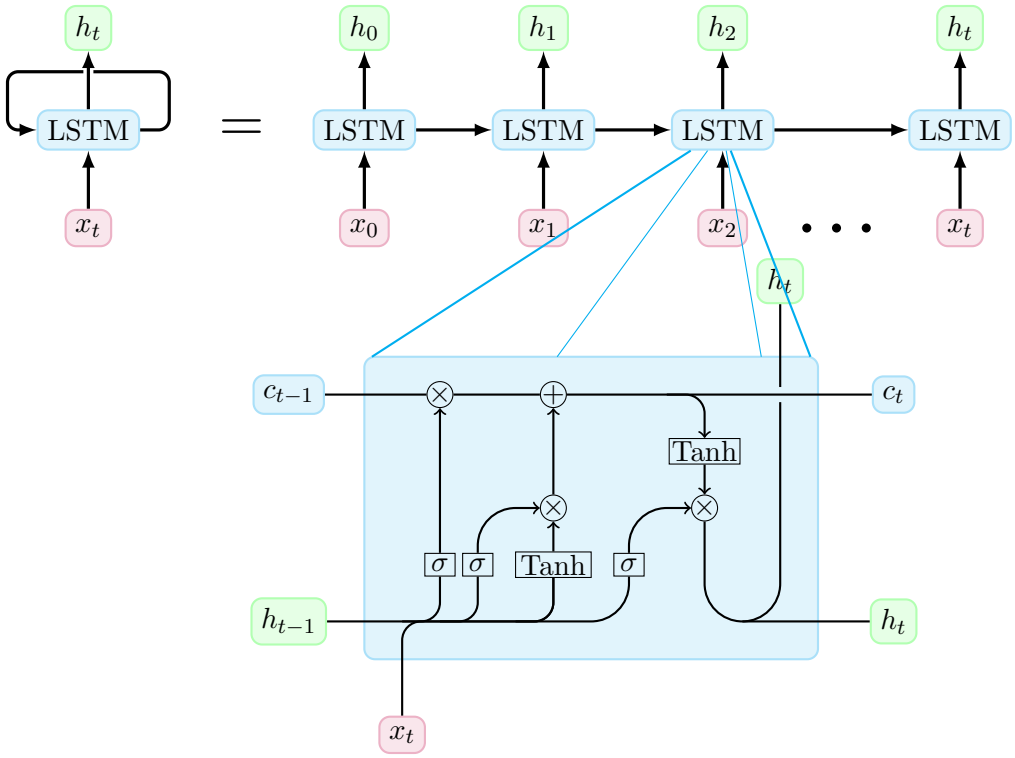

Figure 7: The generic architecture of an LSTM network, with one LSTM cell magnification.

Model	Grouping	Train / Test	Mean rank	Top 1	Top 10	Chapter amount
Closest Neighbour			54	659	2582	4469
Procruste transformation		Train	211	170	929	3128
		Test	225	59	361	1341
LSTM — MSE	40 courses 1058 sessions	Test	1752	0	0	1424
		Train	132	30	306	4021
		Test	1462	1	2	1255
		Train	6	1734	2786	3214

Table 5: Comparative results on the YaleOpenCourseware corpus

Output. The output will be a vector of the same dimensions as the vector representing each chapter. The predicted vector as to be as close as possible in cosine distance to the target vector (vector of the following chapter). This choice may seem surprising, since we could simply see the task as one of classification over the rest of the chapters. Nevertheless, for recommendation purposes and particularly when the corpus grows rapidly and frequently, predicting a vector has the huge advantage of not having to retrain the model each time a resource is added.

Another interesting point is to limit the prediction to only one resource, without any change in the model, we could have predicted to choose not just the next resource but rather the next n with $0 > n >$ number of chapters in the session. Even so, it is an interesting way to investigate, we decide to limit our a prediction to only the next resource.

Preliminary results.

Comparative baselines. To estimate how relevant our results are, we can use simple algorithms to solve our problem. Easy to implement techniques are set as baselines. The first solution is

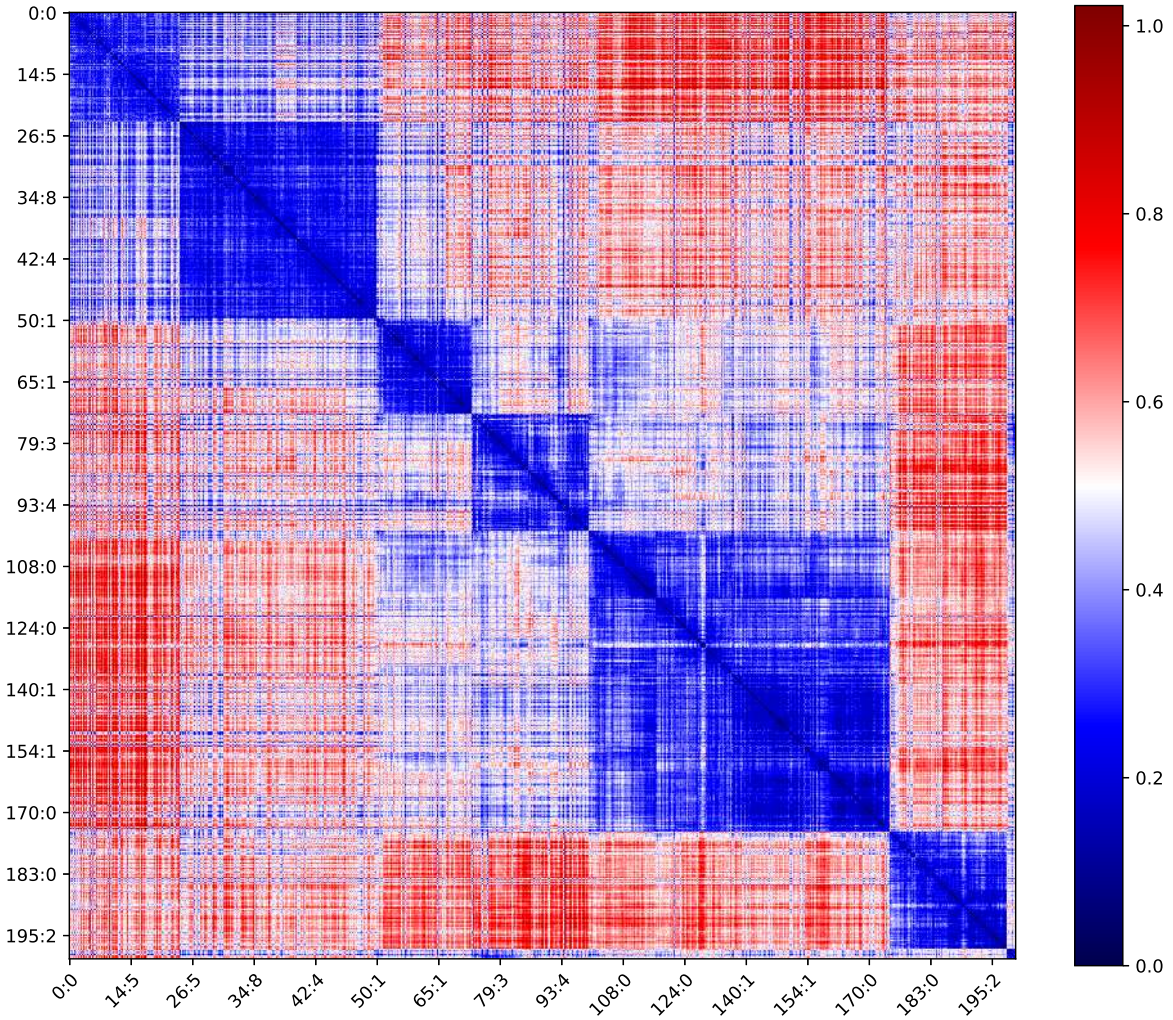

Figure 8: Cosine similarity between chapters vectors of the 200 first sessions.

to predict a chapter by looking at the closest neighbour of the previous one on a cosine distance.

The second solution consists in applying a procrustes transformation on the resources to predict the following ones. We can consider the array of input vectors as the matrix A and the output matrix of the following vectors B. The transition matrix Q of the following expression: $A.Q = B$ can be found by minimizing $||B - A.Q||^2$ subject to $QQ^T = Id$. Rephrasing, we search for the rotation matrix Q which minimizes the cosine distances between a vector and the next one in the session. These two baselines are respectively referenced in the results –Table 5– as closest neighbour and procrustes transformation.

Interpretation. The results show clearly that the LSTM is over-fitting; we hope modifications in architecture like reducing the number of parameters [18], insert dropout [19] or eventually normalization layer [20] will lead us to a more generalized model. This clearly explains the very weak result on the testing set. We note many of the state arts model for recommendation are not able to deal with previously unseen resources.

Despite this point, on the training set these earlier experiments constitute a proof of concept of the capacity of the model to recover the target chapter. More interestingly, the model is also able to attribute a high rank to the targeted chapter even when it is selected as top1, $> 50\%$ of the time in the top and ranked in 23rd position in average. Particularly, the model outperforms both baselines on the training set.

Another interesting point is that *closest neighbour* outperforms Procrustes transformation, this proves that the task can not be reduced to a space rotation.

Perspectives.

To take benefits of the huge amount of data in the database. From our experimental results one of the main weaknesses of the LSTM models is that they are over-fitting to our training set; one way to counter this effect is to provide more data to the model. The first step will therefore be to train the model on the whole database instead of only the YaleSeriesCorpus. Considering the problem as a self-supervised one, the task can remain the same by simply replacing the chapter prediction question presented above by the next chunk prediction using ContinuousDoc2Vec tensors.

To predict the previous chunk. Another interesting path to explore will be to try to predict the previous chunk instead of the following one.

To change text representations. Finally, even if Doc2Vec is a powerful tool to represent documents, it suffers a lot in our application from a lack of interpretability: indeed, the vector is very dense and the dimensions do not represent any interpretable semantic information for humans.

In our case, it would be desirable to be able to interpret the prediction (and longer-term recommendation) made by the model. One way to do this is to use more parsimonious and interpretable representations such as the Wikifier, as input and output of the model. In order to facilitate the learning of such a model, an approach allowing to encode the Wikifier vector into a dense vector and then to decode it afterwards should be considered.

2.2.4 Predicting the missing resource

Principle. In 2018, we worked on the task called "can we find the missing resource?".This consisted in selecting three consecutive resources and removing the second one. This second one was mixed with

a thousand random resources and the task consisted in finding the missing resource.

A long term goal of X5GON is to be able to propose a series of resources rather than a single resource. For this typical question LSTM architectures are naturally relevant, since they are built to deal with sequences as inputs but also as outputs.

Nevertheless, the long sequence predictions also asks the questions "what does the user want to reach ?". Intuitively, the more the recommendation we provide the more the goal you want to reach matters. In order to fill the gap between the current knowledge of a user and his goal, bi-LSTM architectures seem relevant, since they allow to learn not only from the knowledge (previously seen resources) but also from the goal (resources we want to reach).

In the Y1 evaluation, the knowledge was represented by the first resource and the goal by the third one.

As a first step towards long term recommendation, we chose to train a bidirectionnal LSTM [21] on the same task as in Y1. We suppose that a model capable of long term recommendation is *a fortiori* capable of short term recommendations of size 1 in our case.

Insights. We use the same representation as for *Next resource prediction* (Section 2.2.3) for the chapter.

Inputs. In input of the model instead of having only one matrix, we have two matrices, one for the previous chapters and one for the next chapters. More formally, considering a sequence of chapters $c_1, c_2, \ldots, c_i, \ldots, c_{n-1}, c_n$, in order to predict c_i the model will take as inputs two matrix $prev = [c_1|c_2|\ldots|c_{i-1}]$ and $next = [c_{i+1}|\ldots|c_{n-1}|c_n]$.

Output. We apply the same logic as *Next resource prediction* (Section 2.2.3): the model has to allow to predict an embedding vector as close as possible to the targeted one, the recommendation is then done using cosine similarity between the output vector and the vector for each chapter.

Preliminary results.

Comparative baselines. As baselines we designed two methods from Y1; these baselines have not been evaluated using the current architecture, since these baseline methods had been trained to predict the missing resource while the bi-LSTM is trained to predict the missing chapter.

Nevertheless, they give an idea on how well a simple approach can perform on a comparable task.

First, we reuse the method presented in Y1, as a reminder: each session is represented as a bag of concepts. We define *prev* and *next* as the bags of concepts respectively for the previous and the next session. This method predicts the session s which maximizes $|(s \cap prev) \cup (s \cap next)|$, literally the session which has the most concepts which are present in *prev* and *next*. We also tested a variation of this method using the symmetric difference instead of the union of intersections: literally the session which has the most concepts which are present in *prev*, *next* but not in both. Finally, we investigated the impact of weighing down the impact of each concept instead of simply using the cardinality. The results of the symmetric difference technique are not reported in Figure 6, as these were disappointing.

As a second baseline, we consider to simply predict the vector which minimizes the sum of cosine distances with the previous and the next session, rephrased as a formula: $arg\_min_{s \in S} cos(s, prev) + cos(s, next)$. We report two experiment with this approach: one using the WIKIFIER vector and the second using the DOC2VECvector.

Cosine distance approach						
Model				Top 1	Top 10	Support
Doc2Vec				247	677	887
Wikifier (PageRank)				312	727	887
Wikifier (cosine)				301	726	887
Tf-Idf (1-grams)				247	677	887
Tf-Idf (2-grams)				337	743	887
Year 1 approaches						
Model			Weighted	Top 1	Top 10	Support
Union Wikifier (PageRank)			False	94	431	887
Union Wikifier (PageRank)			True	188	612	887
Union Wikifier (cosine)			False	71	367	887
Union Wikifier (cosine)			True	114	526	887
Union Tf-Idf (1-grams)			False	226	623	887
Union Tf-Idf (1-grams)			True	272	598	887
Union Tf-Idf (2-grams)			False	262	711	887
Union Tf-Idf (2-grams)			True	300	707	887
Bi-LSTM						
Model	Grouping	Train / Test	Mean rank	Top 1	Top 10	Support
Bi-LSTM	40 courses	Train	6	1734	2786	3214
		Test	1462	1	2	1255

Table 6: Summarize results for predict missing task on YaleOpenCourseware corpus

Interpretation. We observe from the experiments that the Tf-Idf (in particular in the bi-gram version) is the best representation for both baselines. When we uses Wikifier in our experiments it was beneficial to use the PageRank score rather than the cosine score: we assumed the Wikifier score in this unweighted version is not the same for PageRank and cosine (these are surprising results since in the unweighted version the score is not used). To explain in more depth, the best results for cosine were obtained considering all returned concepts whereas the best results for PageRank were obtained considering only the top 1/7th of these concepts.

About the use of weight, the weighted version of a metric always improved the number of top1 but reduced the number of top10 in the case of the Tf-Idf.

To summarize the two first parts of the table, an adhoc approach can reach a score of around $80 - 83\%$ of top10 and $33 - 37\%$ of top1. This observation confirms our previous year results.

For the bi-LSTM, we observed the same problem as with the LSTM in Section 2.2.3; we hope to be able to fix these problems with improvements discussed is this section. We also obverse that the task of predicting the missing lecture/chapter seems to be easier than to predict the next one, which seems logical given that the bi-LSTM has more information for the same prediction. Comparatively with the baseline approach the model is able to reach around $> 86\%$ for the top10 and $> 53\%$ for the top1 on a similar task.

Perspectives. Several perspectives mentioned for the Section 2.2.3 remain valid here. In order to improve our comparisons an interesting step will be to evaluate the baseline on the chapter predictions.

3 User Analytics

The data collected through CONNECT-SERVICE is able to track user activity. Nevertheless, the data is weak in the following two senses:

- Only data about the web-pages is available: not about the actual final resources which were used/consumed by the user; in other words we can know a user has visited a web-page which contains links to OER files, but we cannot know which of these has been consumed.
- Date-stamps only allow to know when a given web-page was accessed; but we don't have quality information as to how the user has engaged with the material.

The ambition is to build, for each user a topic vector which can be compared with the topic vector of actual resources.

3.1 Building a topic vector for a user

The goal is to build a topic vector (in line and comparable to a WIKIFIER vector for content (see Section 2.1).

In other words, we are given some surface information about the sequence of pages viewed by a user, each page possibly linked with various OER, can we build a profile of the user which will be embedded in the same topic space as the one in which the resources are represented?

Technically, we model this problem as follows:

Through CONNECT-SERVICE, a user will leave traces of activities. A path (of activities) is composed of items (u_i, d_i), meaning that she was on page u_i at date d_i. Therefore a user will be associated to a path $p = \langle (u_0, d_0), (u_0, d_0) \dots (u_m, d_m) \rangle$. On each web-page u_i some resources are accessible; we denote these by R_i^0,..,$R_i^{n_i}$. With each resource R is associated a vector of concepts, as obtained through the WIKIFIER. We denote this vector as $c(R)$, and each possible concept is a dimension so writing for example $c(R)[k]$ makes sense: this would correspond to the WIKIFIER score of the k^{th} concept for resource R.

And in order to solve the general question, we want to answer the following questions.

1. If we are given a path of resources π (ie, if we know what exact resources have been accessed), and we make the extra hypothesis that the resources have been **consumed**, what does $c(\pi)$ look like? In other words, what are the learner's topics?

2. We consider two consecutive pages u_i and u_{i+1} in some learning path. R_{i+1}^j is a particular resource accessible from page u_{i+1}; the previous web-page u_i contains resources $R_i^0, .., R_i^{n_i}$. What is $Pr(R_i^k | R_{i+1}^j)$, the probability that the user's previous resource was R_i^k?

   We make the assumption that $\sum_{0 \leq s \leq n_i} Pr(R_i^s | R_{i+1}^j) = 1$, ie. that the user was on web-page u_i and did consume one resource.

   This hypothesis may be revisited at a later stage: a user could go through a web-page without consuming a resource.

3. If we suppose that web-page u_i was visited just before web-page u_{i+1}, and that in page u_i resource R_i^j was consumed, what is $Pr(R_{i+1}^k | R_i^j)$, the probability of consuming resource R_{i+1}^k (a particular resource accessible from page u_{i+1}) after R_i^j?

4. Given a sequence π of web-pages, what is the profile (the vector of topics) corresponding to the user? This would be $\sum Pr(\pi).c(\pi)$. Each π is a particular path of resources consistent with the learning path.

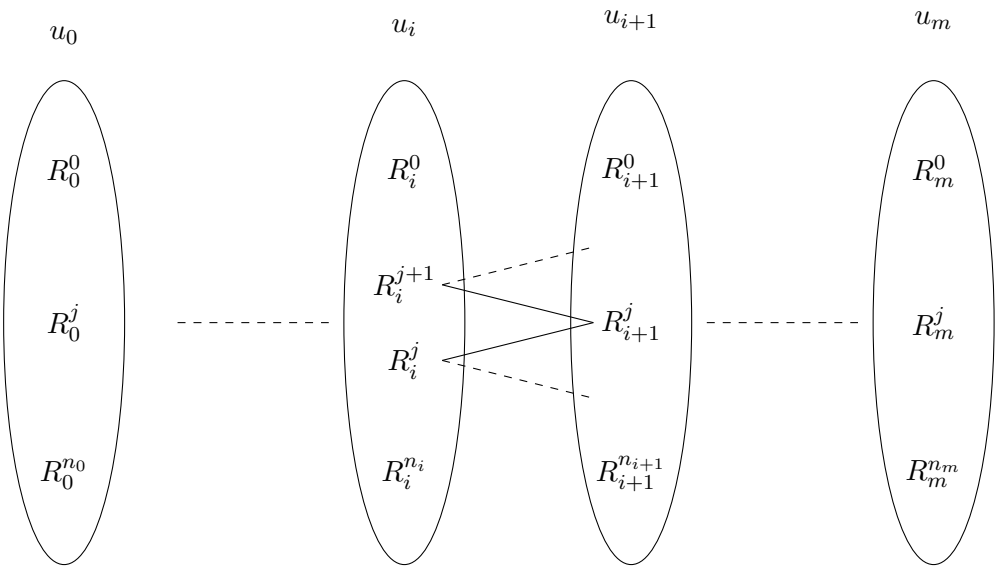

Figure 9: The path of knowledge

Through dynamic programming we provide a positive mathematical answer to all 4 questions. The tests have not been finalized.

We give mathematical answers to these questions in the next two sections.

3.2 Reconstructing learning paths

Let us detail some of the answers to the above questions:

1. Knowledge can be accumulated and summed; a decay function can be applied; a product can be found. Each definition will depend on what we are modelling. We have chosen a simple definition:
$$c(\langle R_0^{k_0} \cdots R_i^{k_i} \cdots R_m^{k_m} \rangle) = \sum_{0 \leq i \leq m} c(R_i^{k_i})$$
With a decay function we generalize this to
$$c(\langle R_0^{k_0} \cdots R_i^{k_i} \cdots R_m^{k_m} \rangle) = \sum_{0 \leq i \leq m} (1-\eta)^{m-i} c(R_i^{k_i})$$
where $0 \leq \eta < 1$.

2. Let us define
$$score(R, R') = c(R) \cdot c(R')$$
where $\cdot$ is the dot product. A cosine distance is an alternative. Then
$$Pr(R_{i+1}^k | R_i^j) = \frac{score(R_{i+1}^k, R_i^j)}{\sum_{0 \leq s \leq n_{i+1}} score(R_{i+1}^s, R_i^j)}$$

3. We are able from the previous point to compute $Pr(R_i^k | R_{i+1}^j)$, so, through Bayes we obtain:
$$Pr(R_{i+1}^k | R_i^j) = \frac{Pr(R_i^j | R_{i+1}^k) \cdot Pr(R_{i+1}^k)}{Pr(R_i^j)}$$

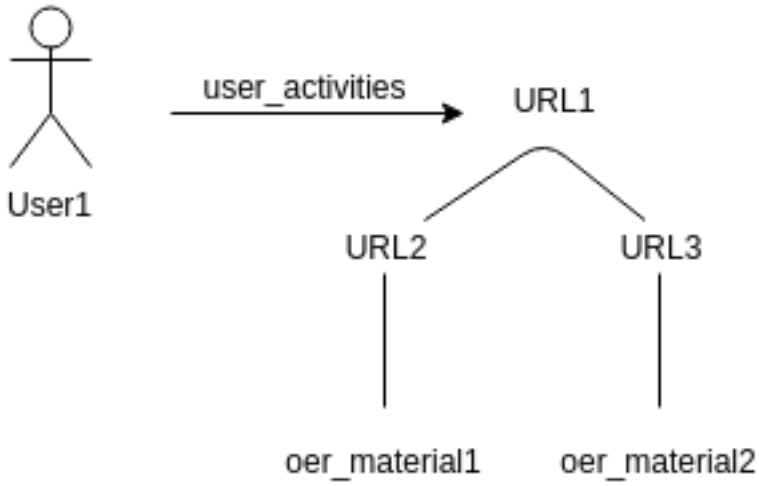

Figure 10: A user's transaction in origin database

$Pr(R_i^j))$ and $Pr(R_{i+1}^k)$ can be estimated to $\frac{1}{n_i}$ and $\frac{1}{n_{i+1}}$ or better to the actual relative popularity of the resources in the page.

3.3 User profiling

The fourth question can receive an answer through dynamic programming. The graph represented in Figure 9 can now be explored.

Let $p = \langle (u_0, d_0), (u_0, d_0) \dots (u_m, d_m) \rangle$ be a path of webpages. A path of resources π *explains* $p = \langle R_0^{k_0} \cdots R_i^{k_i} \cdots R_m^{k_m} \rangle$ if $\forall i \leq m, R_i^{k_i}$ is a resource accessible in page u_i. We denote by Exp(p) the set of all resource paths which explain p.

$$c(p) = \sum_{\pi \in Exp(p)} Pr(\pi) c(\pi).$$

The model, as represented in Figure 9 can be enriched by additional knowledge about how easy the resources are to be found in the page.

3.4 Probabilistic relational models for recommendation

A Probabilistic Relational Model (PRM) is composed of two components: a relational schema of the domain, and a probabilistic model which describes the probabilistic dependencies in the domain. In order to populate our relational schema, we started by studying the existing X5GON project database.

Understanding of the existing database. One limitation of the current database is that in the table `user_activities`, `url_id` does not refer to an `oer_material`. In contrast, it refers to an url that can contain many urls which in their turn refer to `oer_materials`. Figure 10 illustrates an example of a user's transaction in the original database. We can see that the principal URL (URL1) contains other urls (URL2 and URL3) which each one refers to an `oer_material`. To better understanding the distribution of the principal urls with their `oer_materials`, the histogram shown in Figure 11 was constructed based on the 67359 urls consulted by the users in the table `user_activities`. From this histogram, we can notice that only 10000 urls contain one `oer_material`, 8000 urls contain 2 `oer_materials`, 100 urls contain 4 `oer_materials`.

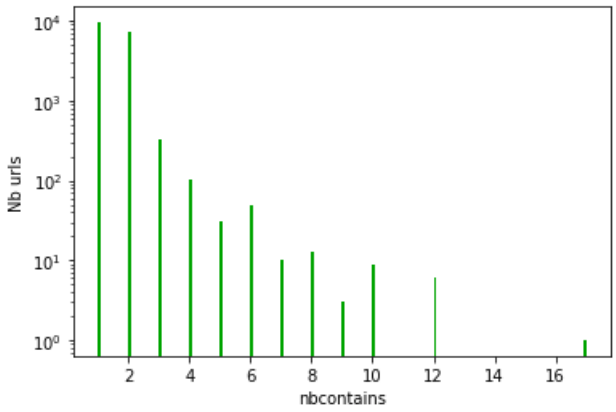

Figure 11: Distribution of the main urls with their `oer_materials`

The relational schema we propose to build a PRM-based recommender system has two entity classes called user and document, and two relationship classes, called `consultation` and `Is-Similar-To`, which represent the relationships document-user and document-document respectively. In order to populate the table `Is-Similar-To`, we make the following hypothesis:

Hypothesis 1 *In the context of our recommender model, we are not interested in proposing documents that are not very similar to the target document. In other words, we are not interested in recording pairs of documents with very weak similarity in table* `Is-Similar-To`.

We chose to represent a document by a vector containing WIKIFIER concepts with their corresponding Page ranks. Then, the similarity between a pair of documents, was computed as the cosine between the corresponding vectors. These similarities corresponding curve is illustrated in Figure 12.

This latter shows that the similarities between 0 and 0.1 appear most often in the data set. However, according to hypothesis 0, we did not take into account the pairs of documents with similarities less than 0.1. After that, we calculated for each document, its associated documents, i.e. its neighbours. Figure 13.(a) shows the distribution of documents according to their neighbours with similarities between 0.1 and 0.4. This figure shows that most of the documents present a significant number of neighbours. In order to lighten our model and the calculations, we kept, in table Is-Similar-To, pairs of documents with similarities greater than 0.4. Then, we built histograms for each of these intervals of similarity: 0.4-0.6 (Figure 13.(b)); 0.6-0.8 (Figure 13.(c)); 0.8-1(Figure 13.(d)). Afterward, we attributed to the similarities which are between 0.4 and 0.6 the class 'low', to the similarities which are between 0.6 and 0.8 the class 'medium', and to the similarities which are between 0.8 and 1 the class 'high'.

Probabilistic Relational Model. In order to build our PRM, we need to define the dependency structure of our model and also its parameters. We propose one first dependency structure in Figure 14 where we define the fact that one first pertinence indicator (direct pertinence) related to one document depends on the number of times this document has been consulted. Besides, we define the indirect pertinence of one document as the weighted sum of the pertinences of its similar documents (where the weight is related to the degree of similarity between both documents).

This indirect pertinence will be used to predict the interesting documents to recommend when one user is reading one target document.

Once the structure of the PRM is defined, we must assign to every node a probabilistic distribution of its domain values given its parents ones. This task can be done by statistical estimation or by

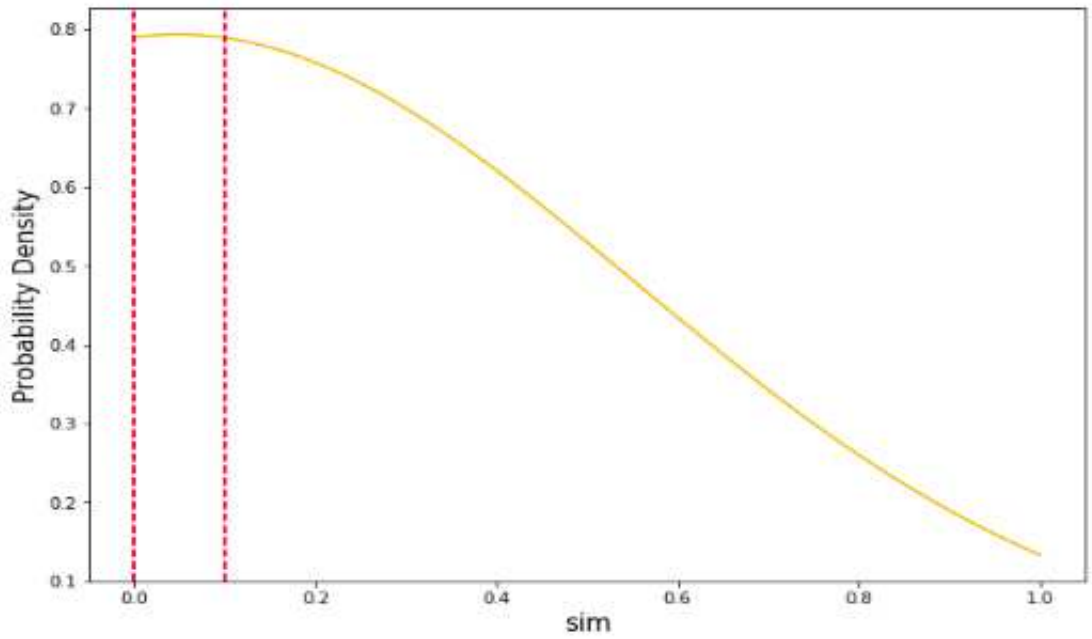

Figure 12: Probability density estimation

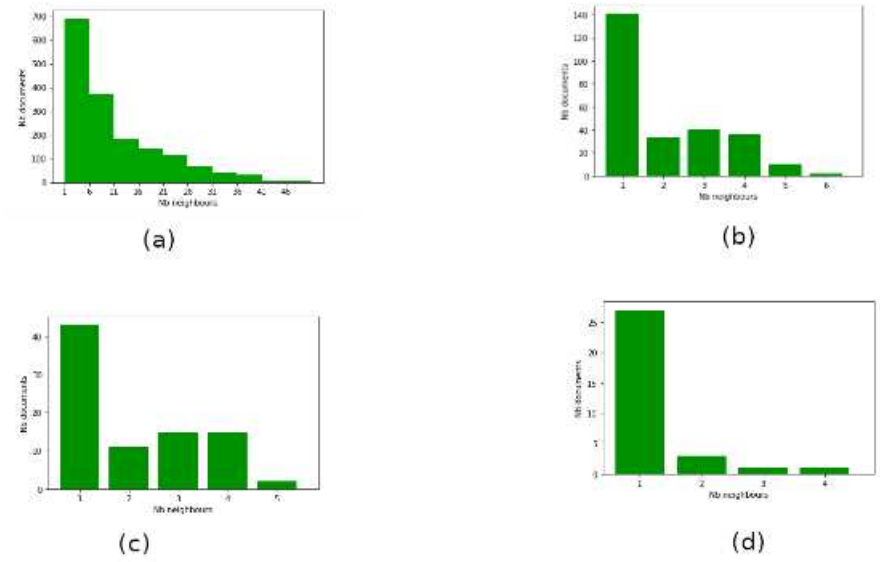

Figure 13: (a) Number of `oer_materials` according to the number of neighbours with $0.1 < sim < 0.4$; (b) Number of `oer_materials` according to the number of neighbours with $0.4 < sim < 0.6$; (c) Number of `oer_materials` according to the number of neighbours with $0.6 < sim < 0.8$; (d) Number of `oer_materials` according to the number of neighbours with $0.8 < sim < 1$;

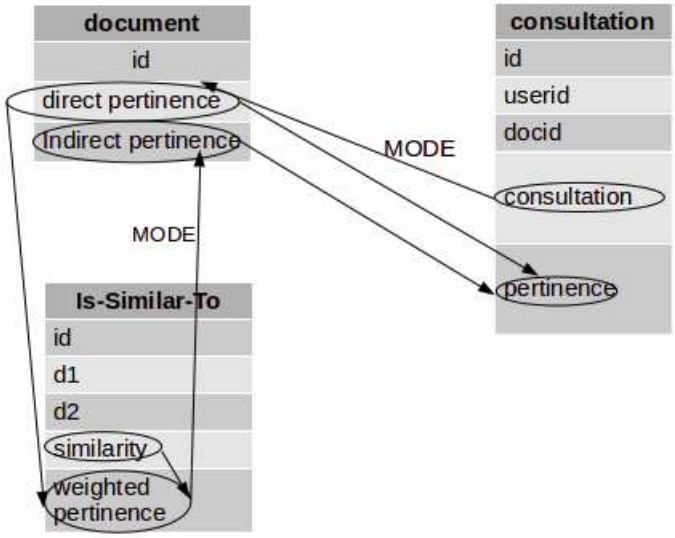

Figure 14: The PRM dependency structure

expertise.

This indirect pertinence will be used to predict the interesting documents to recommend when one user is reading one target document.

Implementation and experiments. Right now, the relational schema and the database have been populated from the X5GON database.

The PRM described above must now be implemented and tested with this database. Results will be compared with the one obtained with the recommender system actually used in X5GON project. We will also be able to learn the structure and/or the parameters of our PRM from the actual database. Our model will also be improved when new data will be available in order to take into account more interesting features from user profile.

4 Accessing the analytics

The analytics and models described in the previous Sections 2 and 3 can be accessed and used in two ways:

- through an API described in Section 4.1,
- through a visual platform described in Section 4.2.

4.1 The API

In this section we explain how the API is designed, how it can be used and what features are offered by the API.

4.1.1 General architecture

The LAM API is a python flask API offering a list of services (explained in Section 4.1.2) accessible by HTTP requests. For the moment, those services are accessible only within the platform (ie. internally). If needed, some of those services could be made accessible externally.

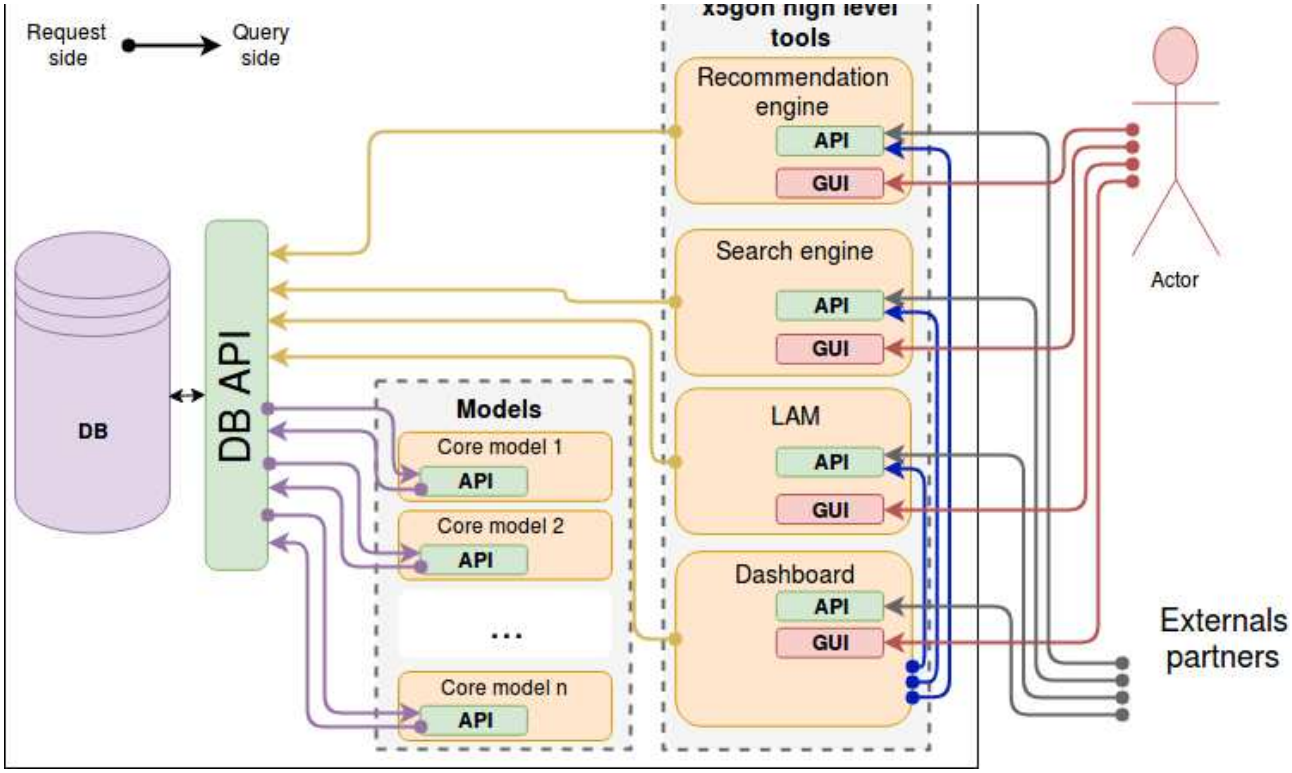

Figure 15: Sharing strategies: interactions between each part of X5GON data platform

The architecture of the API is inspired mostly from our X5GON *sharing strategies proposal* (see Figure 15). In this document we detail the best strategies to share model results regarding the different types of models that we can provide within the consortium. To do that, we envisage a potential architecture in which we show how it will be the relations between the models and the database in which we store data about oers and the related models results.

In summary, models must communicate (fetch and update) with the database only via a `DB API`. For this year, since the `DB API` is not yet fully ready to guarantee such role, we decided to ensure it temporarily via the `LAM API`. Briefly, this ensures the DB connection and the control od requests (fetch, update...) to be sure that we respect the architecture explained in the previously discussed proposal. So the `LAM API` has the following architecture:

- A DB connection layer: ensures the DB connection and offers the appropriate services to manipulate (fetch/update/insert) the data inside the DB.
- A Models connection layer: ensures –via a set of appropriate services– the manipulation(generate/compute in real time/store to DB) of models results (stored locally or in the DB).

The *Models connection layer* offers a set of services exploiting the possible features provided by our models as explained in Section 2.

4.1.2 A list of services

Doc2Vec :
returns the DOC2VEC vector stored in the Dev DB for a given resource.

```
curl -i -H "Accept: application/json"
     -H "Content-Type: application/json"
```

```
-X POST http://localhost:5000/distance/doc2vec/getresourcedoc2vec
-d '{"resource_id": '39435'}'
```

returns the K-Nearest-Neighbours for a given resource basing on the distances between Doc2Vec vectors.

```
curl -i -H "Accept: application/json"
     -H "Content-Type: application/json"
     -X POST http://localhost:5000/getknndoc2vec
     -d '{"resource_id": '39435'}'
```

ContinuousDoc2Vec :
returns the continuousDoc2Vecvector stored in the Development database for a given resource.

```
curl -i -H "Accept: application/json"
        -H "Content-Type: application/json"
        -X POST http://localhost:5000/temporal/continuousdoc2vec
              /getresourcecontinuousdoc2vec
        -d '{"resource_id": '39435'}'
```

Continuouswikifier2order :
returns the logical order for a list of candidates comparing to the principal resource based on their continuousWikifier vectors: given a resource and a list of candidate resources.

```
curl -i -H "Accept: application/json"
        -H "Content-Type: application/json"
        -X POST http://localhost:5000/ordonize/continuouswikification2order
              /batch
        -d '{"id-cwk1": '39435',
             "id-cwkcandidates": ['39435', '39426', '39425', '38657']}'
```

ContinuousWikifier :
returns the ContinuousWikifier vector stored in the development database for a given resource.

```
curl -i -H "Accept: application/json"
        -H "Content-Type: application/json"
        -X POST http://localhost:5000/temporal/continuouswikifier
              /getresourcecontinuouswikifer
        -d '{"resource_id": '39435'}'
```

computes the continuouswikifier vector given the id of a specific resource in the DB. The service can also be used with custom texts through passing the text in the `text` instead of `id-text` parameter in the request.

```
curl -i -H "Accept: application/json"
        -H "Content-Type: application/json"
        -X POST http://localhost:5000/temporal/continuouswikifier
              /continuouswikification
        -d '{"id-text": '39435'}'
```

Difficulty :
gives a difficulty score based on the number of characters per second, for a given resource from the DB.The service can also be used with custom texts through using `length` instead of the `id-length` parameter in the request.

```
curl -i -H "Accept: application/json"
        -H "Content-Type: application/json"
        -X POST http://localhost:5000/difficulty/charpersec
        -d '{"id-length": '39435'}'
```

gives a difficulty score based on the number of wikipedia concepts per second, for a given resource from the database. The service can also be used with custom texts through using `length/wk` instead of `id-length/id-wk` parameters in the request.

```
curl -i -H "Accept: application/json"
        -H "Content-Type: application/json"
        -X POST http://localhost:5000/difficulty/wikification2conpersec
        -d '{"id-length": '39435', "id-wk": '39435'}'
```

gives a difficulty score based on a calculation of the most technical keywords in a Tf-Idf vector, for a given resource from the DB. The service can also be used with custom texts through using `tfidf` instead of `id-tfidf` parameters in the request.

```
curl -i -H "Accept: application/json"
        -H "Content-Type: application/json"
        -X POST http://localhost:5000/difficulty/tfidf2technicity
        -d '{"id-tfidf": '39435'}'
```

Predict_missing :
gives the most probable resource from a list of candidates, for a given resource, `previous` and `after` resources from the database.

```
curl -i -H "Accept: application/json"
     -H "Content-Type: application/json"
     -X POST http://localhost:5000/missingresources/predictmissing
     -d '{"previous": '39435', "after": '7',
          "candidates": ['39435', '9', '7', '7']}'
```

Text2phrase :
gives the phrased text stored in the database, for a given resource from the database.

```
curl -i -H "Accept: application/json"
        -H "Content-Type: application/json"
        -X POST http://localhost:5000/preprocess/text2phrase
                /getresourcephrasedtext
        -d '{"resource_id": '39435'}'
```

gives the phrased text for a given resource from the database. The service can also be used with custom texts through using `text` instead of `id-text` parameter in the request.

```
curl -i -H "Accept: application/json"
        -H "Content-Type: application/json"
        -X POST http://localhost:5000/preprocess/text2phrase/text2phrase
        -d '{"id-text": '39435'}'
```

Text2processedtext :
gives the processed text stored in the DB, for a given resource from the DB.

```
curl -i -H "Accept: application/json"
        -H "Content-Type: application/json"
        -X POST http://localhost:5000/preprocess/text2processedtext
                /getresourceprocessedtext -d '{"resource_id": '39435'}'
```

removes the stop words for a given resource from the DB. The service can also be used with custom texts through using `text` instead of `id-text` parameter in the request.

```
curl -i -H "Accept: application/json"
        -H "Content-Type: application/json"
        -X POST http://localhost:5000/preprocess/text2processedtext
                /removestopwords
        -d '{"id-text": '39435'}'
```

lemmatizes a given text of a given resource from the DB. The service can also be used with custom texts through using `text` instead of the `id-text` parameter in the request.

```
curl -i -H "Accept: application/json"
        -H "Content-Type: application/json"
        -X POST http://localhost:5000/preprocess/text2processedtext/lemmatize
        -d '{"id-text": '39435'}'
```

lemmatizes and removes unwanted words (keeps only the nouns, verbs and adjectives) for a given text of a given resource from the database. The service can also be used with custom texts through using `text` instead of the `id-text` parameter in the request.

```
curl -i -H "Accept: application/json"
        -H "Content-Type: application/json"
        -X POST http://localhost:5000/preprocess/text2processedtext
                /lemmatizermvunwanted
        -d '{"id-text": '39435'}'
```

Tf-Idf :
returns the Tf-Idfvector stored in the DB, for a given resource from the DB.

```
curl -i -H "Accept: application/json"
        -H "Content-Type: application/json"
        -X POST http://localhost:5000/distance/text2tfidf/getresourcetfidf
        -d '{"resource_id":"39435"}'
```

computes the Tf-Idf vector for a given list of resources from the DB. The service can also be used with custom texts through using `texts` instead of `id-texts` parameter in the request.

```
curl -i -H "Accept: application/json"
        -H "Content-Type: application/json"
        -X POST http://127.0.0.1:5000/distance/text2tfidf/tfidfbyngrams
        -d '{"id-texts":"[39435,66962,34547]"}'
```

Tr2text :
gives the plain text from a transcription text stored in the DB, for a given resource from the DB. The service can also be used with custom texts through using the `text` instead of the `id-text` parameter in the request.

```
curl -i -H "Accept: application/json"
        -H "Content-Type: application/json"
        -X POST http://127.0.0.1:5000/preprocess/tr2text/removetags
        -d '{"id-text": '39435'}'
```

Wikifier :
returns the WIKIFIER vector for a given resource from the DB.

```
curl -i -H "Accept: application/json"
        -H "Content-Type: application/json"
        -X POST http://localhost:5000/distance/wikifier/getresourcewikifier
        -d '{"resource_id": '39435'}'
```

computes the WIKIFIER vector for a given resource from the DB. The service can also be used with custom texts through using `text` instead of `id-text` parameter in the request.

```
curl -i -H "Accept: application/json"
        -H "Content-Type: application/json"
        -X POST http://127.0.0.1:5000/distance/wikifier/wikification
        -d '{"id-text": '39435'}'
```

4.1.3 The documentation

A full documentation will be prepared and shared within the consortium. The documentation will list all the accessible services including the possible returned results with the signification of each field.

4.2 Visualization

The section shows how the data returned by the LAM API are exploited in the LAM dashboard.

4.2.1 The graph

The old version of LAM dashboard. A brief reminder about the previous version of the LAM dashboard: the interface allows first to search for a resource using a very simple search engine. A list of matches is returned. When a resource is chosen from the list, a main screen is presented and allows acces to the analytics centred on the chosen resource to be examined.

The navigation can take place:

- by entering a new term in the search engine,
- by single-clicking on any node of the graph: the relevant information concerning this resource appears in the different other areas (provider info, resource meta-data, resource keywords, resource wikipedia concepts),
- by double-clicking on any node of the graph: the chosen node becomes the new current central resource and the main screen is reloaded with a graph containing this new resource as centre.

The focus in this new version The main focus in Y2 was first of all to connect the LAM dashboard (especially the graph) to the X5GON database and feed it with analytics applied on the real data. A second focus was to explore alternative representations showing a resource in relation with its neighbors.

Novelties of this new version. Trying to fulfill the above goals, the LAM dashboard now:

- is connected to the X5GON database on the level of its different sub-elements:
  - LAM API is the back-end engine of this dashboard. Since it is now fully connected to X5GON database, the dashboard is fed always with the fresh data (brute and analytics data),
  - Resource meta-data are fetched directly from the database,
- has a simple search engine that searches in the resources titles in the database,
- Has a neighbours graph with the following features:
  - K-Nearest Neighbours:
    We used the K-Nearest-Neighbours calculated on the X5GON resources. The distance between the resources is computed using the DOC2VECresources vectors stored in the DB.
  - Dimensions reduction:
    To have a better understanding of the resource in its neighborhood, we applied one of the dimensions reduction techniques (PCA: principal component analysis) on the related DOC2VECvectors of the K-Nearest Neighbors of the central resource. We applied that to reduce DOC2VECvectors to 2 dimensions presentable in a 2-D graph.
  - Maximum dimensions to represent:
    Many representations have been tested to visualize best the data in the graph. Finally, we chose a representation using:
    * Node color to indicate the resource difficulty. For difficulty measurement, we used the Kurtosis model applied on the TF-IDF resources vectors stored in the DB.
    * Node size to indicate the resource length.
  - Visual features were added to reinforce the graph readability:
    to show better resource information, some graphical features were added (Figure: 14):
    * an Info-ball appears when hovering the resource. The pop-up contains some useful meta-data about the resource in question,
    * a transparency effect was introduced when hovering the resource, keeping the resource in focus.
    * drag and drop effects are possible with resources to be able to move temporarily a specific resource.
    * a *pulse* effect is added to the central resource to keep it in focus.

4.2.2 Connecting with the real data

An important objective was to fill the LAM dashboard with the real data from X5GON collected during the last 2 years from the different repositories who joined the consortium or other open repositories. And then to replace the first generation of models from the LAM dashboard V1.0 by the new ones. By doing this, the data now used by the API are the most actualized data available.

Here are the numbers:

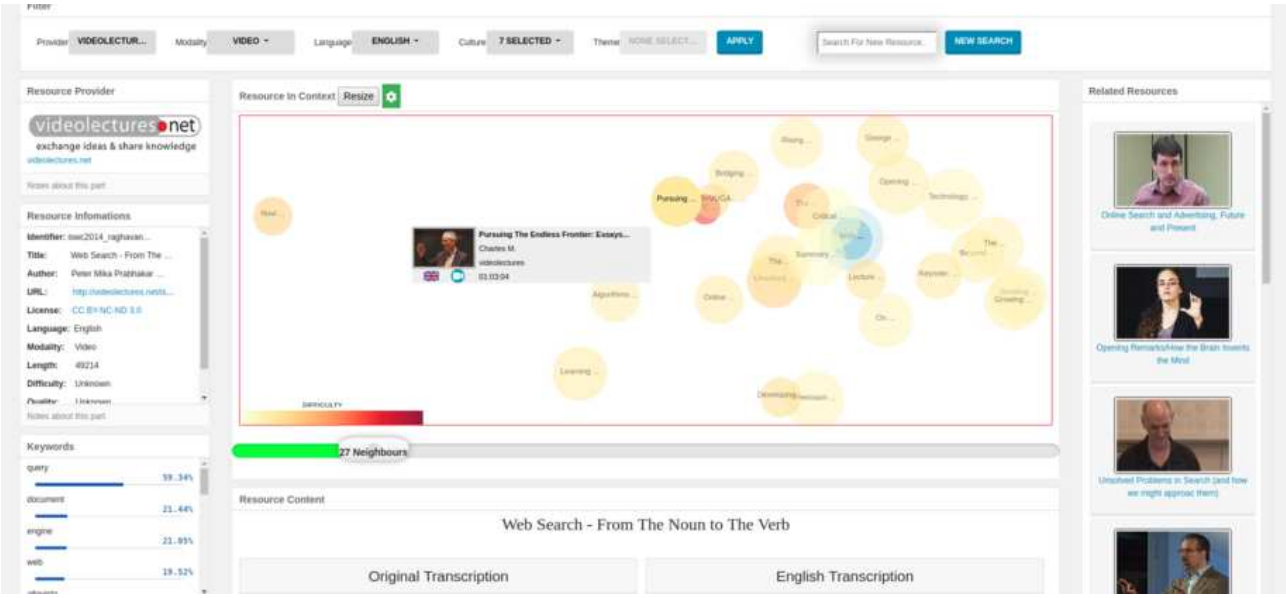

Figure 16: Overview of the LAM dashboard v2

Figure 17: Neighbours graph of specific resources

- 90023 open educative resources,
- 219238 open educative resource contents(transcriptions and translations),
- 313169 different cookies,
- 1176184 user activities,
- 13 open repositories,
- 582630 models results (experiments results),
- 8 models experiments.

For this actual version of the LAM dashboard, we used the resources meta-data and some of the content models (Wikifier, Tf-Idf , Doc2Vec...) stored in the X5GONdatabase. To do that, we prepared a script (connection to the DB via an `ssh` tunnel) to give us the ability to work on the updated data regularly.

Now the LAM API and dashboard are fed directly from the X5GON database (the dev DB).

This enabled us to view and apply our visualization approaches on the real data and see the limits of our models in real time.

5 Acquiring the Data

Even if the acquisition of the data is not strictly speaking part of this work-package, we indicate here some aspects, because of their implications for the LAM.

5.1 Corpus crawling

Different corpora containing open educational resources were identified and crawled separately: in all cases these corpora provide higher quality annotations which can be used for supervised learning algorithms.

The YaleOpenCourseware corpus. The YaleOpenCourseware project provides free and open access to a selection of introductory courses taught in English by distinguished teachers and scholars at Yale University. Each course is composed by a series of lectures (called sessions); the handmade transcriptions and chapter division are available for each session.

The corpus crawled from the project website: https://oyc.yale.edu/ contains 40 courses and 1058 sessions with an average of 26.45 ± 4.8 sessions/course. The corpus, the code and all information can be found at https://gitlab.univ-nantes.fr/connes-v/yaleocw-corpus.

The Amazoninspire corpus. Amazon Inspire is an open collaboration service that helps teachers to easily discover, gather, and share quality educational content with their community. Teaching materials cover a large variety of subjects including Mathematics, English Language Arts, Science, Social Studies and Arts. Resources are provided for students in grades K-12, in various formats (doc, docx, pdf, pptx, open links, etc.). The corpus crawled from the project website *https://www.amazoninspire.com/* includes thousands of resources.

CK12 digital textbooks. Due to the lack of good-quality educational content, additional resources have been crawled manually from the CK12 platform. CK12 provides a library of free online textbooks, videos, exercises, flashcards, and real world applications for over 5000 concepts in multiple fields. Notably, grades are provided for each of resources.

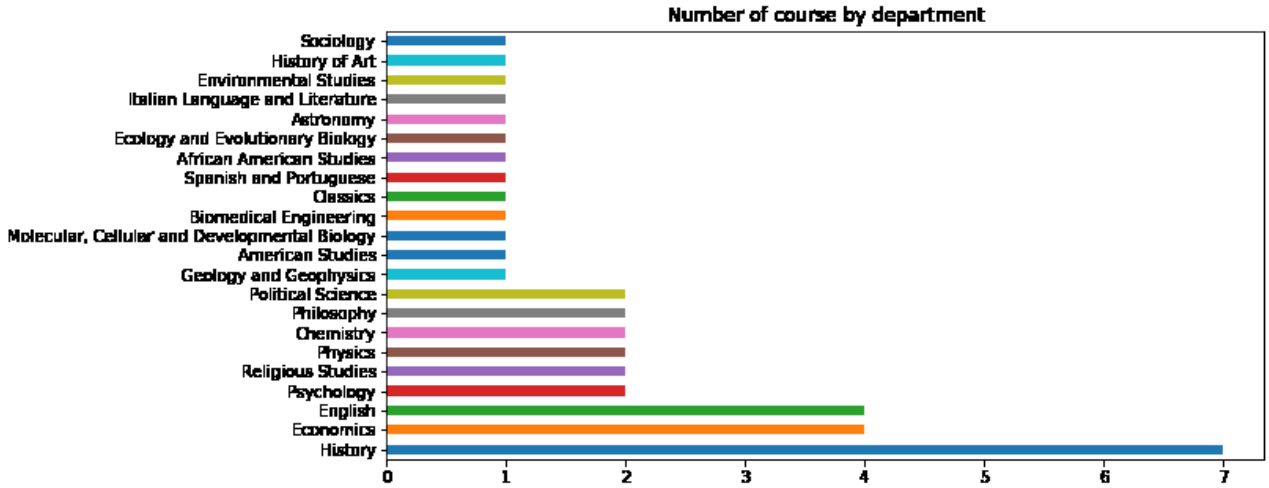

Figure 18: Distribution of courses in department in YaleOpenCourseware corpus

5.2 Upload API

The data acquired on the Amazon Inspire platform were added to the project database in order to benefit from the transcription and translation services of UPV. The resources were successfully processed within a few days. This shows that the upload API is a powerful tool to quickly add new resources to the project. In the future, we hope that this facilitates the incorporation of new partner directories.

5.3 Moodle Plugin

X5GON Moodle plugin ensures the integration of the CONNECT-SERVICE automatically. CONNECT-SERVICE is activated for the selected Moodle pages and resources. The selection depends on the plugin parameters and the resources meta-data (only open resources are connected to X5GON). This is ensured through the following functionalities.

5.3.1 Connecting the selected resource

CONNECT-SERVICE is activated on the courses pages and their sub-pages and notifies X5GON about any access taking place on one of these pages if the following conditions are met:

1. Course category: attribute `course.category` must be set in the plugin configuration.
2. Course visibility: attribute `course.visibility` should be equal to 1 or equal to another administrator defined value.
3. Course enrolment type: a course should have at least an `enroll type` equal to "guest" and attribute `enrol.password` should be empty, or `enroll` should have another administrator defined value.

5.3.2 Acquiring user activities traces

The plugin sends information to X5GON about user activities when accessing OERs. Here is the listed user activities which are considered:

- actions made on any type of Moodle modules contained in the course page,

- actions (pause/play) made on media players (external or internal resources) contained on any type of Moodle modules in the course page,
- actions on Youtube media players.

The information is sent only if the criteria described in the next 2 paragraphs are met:

OER Criteria

1. Course category: attribute `course.category` should be set in the plugin config.
2. Course visibility: attribute `course.visibility` should be equal to 1 or equal to another administrator defined value.
3. Course enrolment type: course should have at least an enroll type equal to "guest" and attribute `enrol.password` should be empty, or enroll should have another administrator defined value.
4. Course module visibility: attribute `module.visibility` should be equal to 1 or equal to another administrator defined value.
5. Course module availability: attribute `module.availability` should be empty or equal to another administrator defined value.
6. File license: attribute `file.license` should be any one of the Creative Commons (CC) licenses.

User Consent Depending on the Moodle version, the plugin presents the user with the X5GON project privacy policy, if consistent with the running Moodle implementation. As a result, each user will be asked to give his consent. If a user doesn't give his consent, no activities trace is sent to X5GON. This is asked to all types of Moodle users: registered and guest users.

5.3.3 User activity information

The following table contains all information sent by connect-service to the X5GON server. Details are given in the appendices section.

5.4 Accessing the resource meta-data via the Data API

The data API web-service is deployed to respond to the external requests about OERs meta-data. It is provided to communicate external parties cleanly with Moodle and avoid going through the Crawler to get information about OERs.

This data API offers 3 main web-services:

- `oerinfos` returns the meta-data about a specific Moodle module and its related attached final files,
- `courseoers` returns the meta-data about a specific course and its related Moodle modules,
- `oerslist` returns the meta-data of all OER Moodle modules which can be found in Moodle classified by course.

5.5 How to install the X5gon Moodle plugin

The X5GON Moodle plugin is easy to install. For that, a documentation is proposed. Once installed, a systems administrator can configure it. We provide here some useful links that can help for installing or to get extra details about the plugin:

- Latest version of the plugin:
  https://gitlab.univ-nantes.fr/x5gon/x5gonmoodleplugin-prod/tree/master/latest/src
- Latest full documentation document:
  https://gitlab.univ-nantes.fr/x5gon/x5gonmoodleplugin-prod/tree/master/latest/documentation
- Latest dissemination document:
  https://gitlab.univ-nantes.fr/x5gon/x5gonmoodleplugin-prod/tree/master/latest/dissemination
- Latest demo video:
  https://gitlab.univ-nantes.fr/x5gon/x5gonmoodleplugin-prod/tree/master/latest/demo

5.6 Future work including Wordpress

The CONNECT-SERVICE is a crucial tool for the X5GON project. The Moodle plugin developed here (see Section 5.3) is now adapted to keep track of the important video use activity data. But even if Moodle is the dominating LMS around the world, Universities and Ministries who have OER repositories tend to prefer using alternative systems.

One such configuration is Wordpress used by several repositories. It is proposed to work on building a plug-in supporting the CONNECT-SERVICE for Wordpress.

6 Storing and sharing the data

How to store and share the data is a critical question for the project. Not only does it have technical consequences, but it also modifies the work organization and the way researchers and developers will share ideas. For year 2 we chose to set up a database architecture able to host the content data, the user data and their respective features. A particular attention was given on the internal sharing of features.

6.1 The database proposal

The database (Figure 19, page 39) is composed of a public part and a private part: the public part is designed to be accessible by the external partners of the project while the private part is reserved for the internal partners. The public part contains all the content related data and all the public features; these features are stored in the *features_public* table. The rest of the data are stored in the private part, in particular user-related data, private features (table *features_private*) and in progress features provided by the partner. The last ones are saved in the three tables: *tools*, *experiments*, *experiment_results*; these tables are particularly important to facilitate the feature sharing between the partners, the comparison or evaluation of different models and for –on the long term– AB-testing purpose. We care about storing in the database only the features that are not efficiently computable on the fly. More information and thoughts about proposal strategies to feature sharing can be found in the document **About inserting and using models in the project database**.

A more detailed explanation of each table can be found at **Database scheme**.

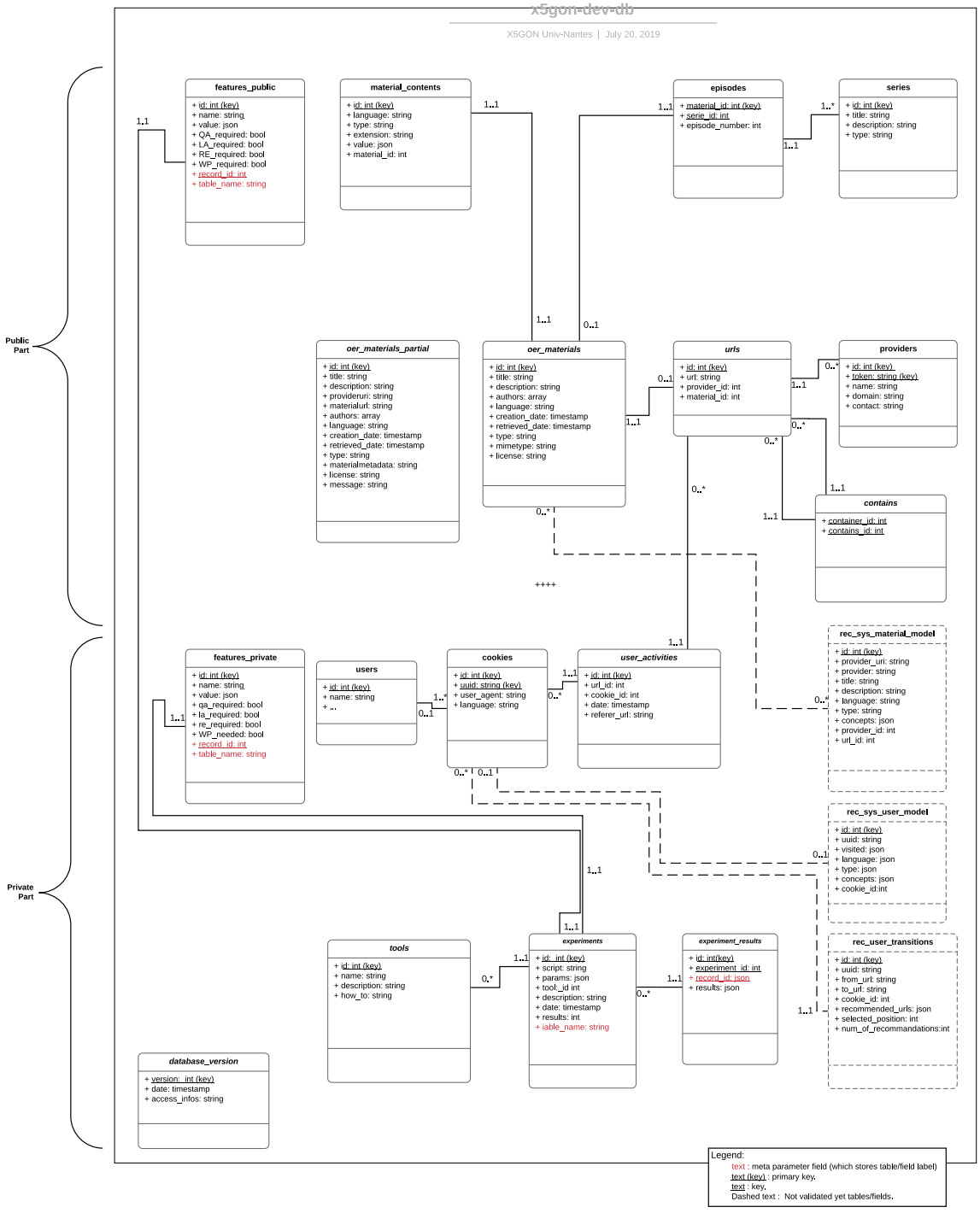

Figure 19: Proposed database architecture

6.2 Development and Production

Since the database is also used as a platform of experiments for all partners, it will be modified or altered very often. To reduce the risk of a human mistake and in order to have the most optimized database as possible for the production, we duplicate the current database. The first copy keeps the original architecture and is dedicated to the development. The second one is amputated of the experiment's tables and is dedicated to production.

6.3 Sharing strategies

The two databases are hosted by Posta servers, and all partners can access the database by `ssh` through a dedicated port.

The discussions and work which led to the database had the added advantage of better linking together the efforts made by each team. The technical choice (Postgres, SQL) was discussed in common: more dynamic choices were evaluated but the proposed choice allowed better harmonization.

7 Conclusion

In this report we have presented the API which computes different models, the tools developed in order to observe and understand these models, and preliminary work which should lead to new models added in year 3.

Is still missing at this point a larger set of visualizations allowing to let different target audiences understand the benefits of using the API. Some of this work is purely technical. Some of it is more political: what are the key arguments to convince:

- a policy maker to adopt X5GON,
- a new repository to see the benefits of joining the growing group of partners,
- a developer to avoid building new models since he can use those we have provided,
- a researcher to invent new ideas based on the possibilities offered by the APIs.

8 Appendices

8.1 Information sent by Connect-Service in the X5gon Moodle plugin

Here is a table summarizing the full information sent by X5GON moodle plugin:

Attribute	Type	Description
x5gonValidated	Boolean	Notifies if the user is validated by the X5GON platform
dt	String	The URI date-time
rq	String	The URL from which the request was sent
rf	String	The referrer URL
cid	String	The provider token generated by the X5GON platform
providertype	String	LMS type: Moodle or another known LMS or else
title	String	LMS type: Resource title in Moodle
description	String	Resource Moodle module description
author	String	Resource Moodle module author
language	String	Resource language
creation_date	String	Resource Moodle module creation date
type	String	Resource Moodle module type
mimetype	String	Resource mimetype if it exists
license	String	Resource license
resurl	String	Resource URL
resmdlid	String	Resource Moodle ID
residinhtml	String	Resource html-id in the page
resurlinpage	String	Resource exact url in the page
rescrstitle	String	Resource course title
rescrsid	String	Resource course ID
rescrssum	String	Resource course summary
rescrslang	String	Resource course language
rescrsctg	String	Resource course category
rescrsctgdesc	String	Resource course category description
mdaccess	String	Access type: `yes` if it is an access to a media resource
mdaction	String	User action type on media resource: `Play/Pause`
mdsrc	String	Media resource source
mdduration	String	Media resource total duration
mdactiontime	String	User action timestamp % resource duration

Table 7: The information sent by the connect service in X5GON Moodle plugin

8.2 About strengths and weaknesses of the Wikifier

In this appendix we present some experimentation done in order to test the robustness of the WIKIFIER and some remarks collected over time on its usage. This section is not a rigorous scientific scientific work but rather a collection of technical observations.

Resilience to translation. In our project, the WIKIFIER is often used on a translated text, a transcribed text or even a transcribed and translated one. It seemed relevant to check not only the effectiveness of the WIKIFIER on spoken language but also its resilience to translation.

In order to test the robustness against translation, we built the following process:

- Select an written English text;
- Extract Wikipedia concepts;
- Automatically translate this text in French (DeepL translation);

Rank	Source	Translation
1	Computing Machinery and Intelligence	Computing Machinery and Intelligence
2	Turing test	Definition
3	Causality	Game theory
4	Mind	Free will
5	Machine	Mind
6	Computer	Machine
7	Statistics	Artificial intelligence
8	Survey methodology	Survey methodology
9	Signals intelligence	Statistics
10	Teleprinter	Human
11	Interrogation	Teleprinter
12	Alan Turing	Interrogation
13	Computer science	Alan Turing
14	Probability	Computer science
15	The Imitation Game (play)	Probability
16	Syntax	Semantics
17	Gallup (company)	Gallup (company)

Table 8: Main concepts for the *Computing Machinery and Intelligence* example

- Translate automatically the French in inverse way;
- Extract the Wikipedia concepts;
- Compare the two sets of Wikipedia concepts extracted.

We ran these experiment over ten texts of various origin, style and subject. These experiments show that the two sets of concepts are not the same, and even if both sets of concepts contain relevant concepts for the documents, a few concepts (around 10) are lost during the process, with some being high-ranked concepts (see Table 8). The translation seems to lead the Wikifier to replace specific concepts by more general ones. As an illustration, these are the concepts extracted from the "Imitation game" section of Alan Turing's work "Computing machinery and intelligence" [22]:

Common concepts : 'Mind', 'Gallup (company)', 'Statistics', 'Probability', 'Survey methodology', 'Syntax', 'Semantics', 'Computing Machinery and Intelligence', 'Teleprinter', 'Alan Turing', 'Interrogation'

Lost concepts : 'Computer', 'Turing test', 'Causality', 'The Imitation Game (play)', 'Military intelligence', 'Computing'

New concepts : 'Free will', 'Game theory', 'Artificial intelligence', 'Computer science', 'Thought', 'Machine', 'Definition', 'Human'

Resilience to preprocessing Even if preprocessing such as removing stop words, lemmatization or part of speech filtering are beneficial in many natural language processing tasks, in the case of the Wikifier these preprocesses can worsen the concepts extraction, since it is dependent of matching directly sub-sentences in the text. An easy solution to mitigate these possible negative effects of preprocesses on the Wikifier, is to simply use the Wikifier on the raw text. Nevertheless, we will investigate these effects based on the assumption that the effects produced by preprocessing will probably be close to those caused by text extraction from pdf or ocr.

Rank	Source	Stop words	Lemmatization	Part of speech filtering
1	Computing Machinery and Intelligence	Computing Machinery and Intelligence	Definition	Game try
2	Turing test	Turing test	Game	Game
3	Causality	Alan Turing	Survey	Definition
4	Mind	Causality	Mind	Teleprinter
5	Machine	Teleprinter	Knowledge	Object (philosophy)
6	Computer	Machine	Object	Interrogation
7	Statistics	Survey methodology	Teleprinter	Semantics
8	Survey methodology	Computer	Statistics	Machine
9	Signals intelligence	Interrogation	Machine	Survey methodology
10	Teleprinter	Game try	Human	Intelligence
11	Interrogation	The Imitation Game (play)	Typewriter	Mind
12	Alan Turing	Communication	Interrogation	Computing
13	Computer science	Statistics	Communication	Statistics
14	Probability	Computing	Writing	Gender
15	The Imitation Game (play)	Probability	Semantics	Communication
16	Syntax	Gallup (company)	Nonverbal	Typewriter
17	Gallup (company)	Military intelligence	Intelligence	Human

Table 9: Concepts after preprocessing.

We followed a similar pipeline as that used for our translation experiments:

- Select an written English text;
- Extract Wikipedia concepts;
- Preprocess;
- Extract the Wikipedia concepts;
- Compare the two sets of Wikipedia concepts extracted.

We preprocessed using the Spacy python library:

1. Stop words removing,
2. Lemmatization,
3. Part of speech filtering to only keep nouns, adjectives and verbs.

The results are given in Table 8.2.

The example on *Computing Machinery and Intelligence* shows us an interesting resilience of the Wikifier against preprocessing, furthermore we observe the same biases as on our experiments on the effect of translation, specific concepts are lost for the benefits of more general ones. Despite that, the Wikifier shows a good resilience against preprocessing.

References

[1] G. Leban J. Brank and M. Grobelnik. Annotating documents with relevant wikipedia concepts. 2017.

[2] Gerard Salton and Michael J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA, 1986.

[3] Gerard Salton, Edward A. Fox, and Harry Wu. Extended boolean information retrieval. *Commun. ACM*, 26(11):1022–1036, November 1983.

[4] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, 24(5):513 – 523, 1988.

[5] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013.

[6] Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents. *CoRR*, abs/1405.4053, 2014.

[7] Eva Lindström. *Language complexity and interlinguistic difficulty*, page 217–242. 01 2008.

[8] Kaius Sinnemäki. Language universals and linguistic complexity : Three case studies in core argument marking. 2011.

[9] Michael Heilman, Kevyn Collins-Thompson, and Maxine Eskenazi. An analysis of statistical models and features for reading difficulty prediction. In *Proceedings of the Third Workshop on Innovative Use of NLP for Building Educational Applications*, EANL '08, pages 71–79, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[10] Luo Si and James P. Callan. A statistical model for scientific readability. In *CIKM*, pages 574–576, 2001.

[11] Michael Heilman, Kevyn Collins-Thompson, Jamie Callan, and Maxine Eskenazi. Combining lexical and grammatical features to improve readability measures for first and second language texts. pages 460–467, 01 2007.

[12] Sarah E. Petersen and Mari Ostendorf. A machine learning approach to reading level assessment. *Computer Speech and Language*, 23:89–106, 01 2009.

[13] Kathleen M. Sheehan, Michael Flor, and Diane Napolitano. A two-stage approach for generating unbiased estimates of text complexity. In *Proceedings of the Workshop on Natural Language Processing for Improving Textual Accessibility*, pages 49–58, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

[14] Wikipedia contributors. Flesch–kincaid readability tests — Wikipedia, the free encyclopedia, 2019. [Online; accessed 15-July-2019].

[15] Paul Nowak. *What Is the Average Reading Speed?* 2018.

[16] J. J. A. Moors. The meaning of kurtosis: Darlington reexamined. *The American Statistician*, 40(4):283–284, 1986.

[17] Gensim. `https://radimrehurek.com/gensim/`.

[18] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *CoRR*, abs/1611.03530, 2016.

[19] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

[20] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization. *arXiv e-prints*, page arXiv:1607.06450, Jul 2016.

[21] Mike Schuster and Kuldip K. Paliwal. Bidirectional recurrent neural networks. *Signal Processing, IEEE Transactions on*, 45:2673 – 2681, 12 1997.

[22] A. M. TURING. I.—Computing Machinery and Intelligence. *Mind*, LIX(236):433–460, 10 1950.

[23] Davor Orlic. D7.1: Website. Technical report, X5gon project, M1, 2018.

[24] Erik Novak. D2.1: Requirements & Architecture Report. Technical report, X5gon project, M6, 2018.

[25] Stefan Kreitmayer. D4.1: Initial prototype of user modelling architecture. Technical report, X5gon project, M6, 2018.

[26] D1.1: Quality assurance models. Technical report, X5gon project, M12, 2018.

[27] D1.2: Report on selected and evaluated quality assurance models. Technical report, X5gon project, M12, 2018.

[28] D3.1: Learning Analytic Engine 2.0. Technical report, X5gon project, M12, 2018.

[29] D6.1: Report of the OER network model and interface design evaluation. Technical report, X5gon project, M12, 2018.

[30] D7.2: First real-world and online community engagement plan. Technical report, X5gon project, M12, 2018.

[31] D8.1: Detailed market analysis. Technical report, X5gon project, M12, 2018.

[32] D9.1: Ethical Data. Management and Data. Management Pla: year 1. Technical report, X5gon project, M12, 2018.

[33] D9.4: First year report. Technical report, X5gon project, M12, 2018.

[34] EMMA. http://project.europeanmoocs.eu/.

[35] poliMedia. `https://media.upv.es/#/catalog`.

[36] poliMedia. `https://politrans.upv.es/`.

[37] Review Report. Technical report, X5gon project, November 2018 (M15).

[38] transLectures. `http://www.translectures.eu/web`.

[39] Jorge Civera and Alfons Juan. T36: Final report. Technical report, UPV, 2014.

[40] The MLLP Transcription and Translation Platform (MLLP-TTP). `https://ttp.mllp.upv.es`.

[41] Juan Daniel Valor Miró, Pau Baquero-Arnal, Jorge Civera, Carlos Turró, and Alfons Juan. Multilingual Videos for MOOCs and OER. *Educational Technology & Society*, 21(2):1–12, 2018.

[42] GSC-TUDa: German Speech Corpus by Technische Universität Darmstadt. `https://www.lt.informatik.tu-darmstadt.de/de/data/open-acoustic-models`.

[43] WEBCELEX: The CELEX Lexical Database (English, Dutch and German word features). `http://celex.mpi.nl/`.

[44] TensorFlow. `https://www.tensorflow.org/`.

[45] TED. `https://www.ted.com/talks`.

[46] The RNNLM Toolkit . `http://www.rnnlm.org/`.

[47] Sequitur G2P. `http://www-i6.informatik.rwth-aachen.de/web/Software/g2p.html`.

[48] News Crawl corpus (from WMT workshop) 2015. `http://www.statmt.org/wmt15/translation-task.html`.

[49] Wikipedia. `https://www.wikipedia.org/`.

[50] Europarl Corpus: European Parliament Proceedings Parallel Corpus v7. `http://www.statmt.org/europarl/`.

[51] commoncrawl 2014. `http://commoncrawl.org/`.

[52] REUTERS: Reuters Corpora (RCV1, RCV2, TRC2). `http://trec.nist.gov/data/reuters/reuters.html`

[53] Tatoeba. `https://tatoeba.org/eng/downloads`.

[54] UPVLC. D2.3.2: Report on final transcription and translation models. Technical report, EMMA, 2015.

[55] M.A. del Agua, A. Giménez, N. Serrano, J. Andrés-Ferrer, J. Civera, A. Sanchis, and A. Juan. The translectures-upv toolkit. In JuanLuis Navarro Mesa, Alfonso Ortega, António Teixeira, Eduardo Hernández Pérez, Pedro Quintana Morales, Antonio Ravelo García, Iván Guerra Moreno, and DoroteoT. Toledano, editors, *Advances in Speech and Language Technologies for Iberian Languages*, volume 8854 of *Lecture Notes in Computer Science*, pages 269–278. Springer International Publishing, 2014.

[56] M.J.F. Gales. Maximum likelihood linear transformations for hmm-based speech recognition. *Computer Speech and Language*, 12(2):75 – 98, 1998.

[57] Geoffrey Hinton, Li Deng, Dong Yu, Abdel rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara Sainath George Dahl, and Brian Kingsbury. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal Processing Magazine*, 29(6):82–97, November 2012.

[58] Albert Zeyer, Patrick Doetsch, Paul Voigtlaender, Ralf Schlüter, and Hermann Ney. A comprehensive study of deep bidirectional lstm rnns for acoustic modeling in speech recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 2462–2466, New Orleans, LA, USA, March 2017.

[59] Xi Chen, Xin Liu, Y. Qian, Mark J. F. Gales, and Philip C. Woodland. Cued-rnnlm — an open-source toolkit for efficient training and evaluation of recurrent neural network language models. *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6000–6004, 2016.

[60] H.W. Hunziker. *Im Auge des Lesers: vom Buchstabieren zur Lesefreude : foveale und periphere Wahrnehmung*. Transmedia, 2006.