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

Grant Agreement Number: 761758
Project Acronym: X5GON
Project title: X5gon: 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
Document Title: D4.3 – Early prototype of recommendation engine
Author(s): Jasna Urbančič, Erik Novak
Contributing partners: JSI
Date:
Approved by:
Type: P
Status: Draft/Final
Contact:

Dissemination Level		
PU	Public	x
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Revision

Date	Lead Author(s)	Comments
20.07.2018	Jasna Urbančič	Initial draft
23.07.2018	Erik Novak	Internal revision
13.08.2018	Colin de la Higuera	First review and feedback

TABLE OF CONTENTS

LIST OF FIGURES

Acronyms	Definitions
OER	Open Educational Resources
API	Application Programming Interface
REST	Representational State Transfer
JSON	JavaScript Object Notation
URL	Uniform Resource Locator
HTML	Hypertext Markup Language

ABSTRACT

The X5GON project is developing a recommender engine to recommend material from multiple OER repositories, allowing the users to gain relevant knowledge from multiple resources. The system will incorporate cross-site, cross-modal and cross-lingual methods and services which will allow us to recommend materials provided in different languages, file formats, and from different sites. In this report we present the initial steps to achieving this goal – the prototype of a recommendation engine.

First we describe the data that was acquired through the X5GON platform. We present the OER repositories from which we acquired the data, as well as basic statistics of the acquired dataset concerning for example the types and languages of the documents.

Next we present the recommendation models we are considering to develop in the project. The models are based on content comparison and user preferences.

Finally, we present the architecture of the recommendation engine prototype and show the initial results. We also present a way to embed the recommendation engine results on a third party OER portal.

The cascade model followed by X5GON means that the first version of the recommendation engine does not depend on work done in parallel in other work packages: this work will be included in the future versions.

1 INTRODUCTION

The main objectives of this project is to create a cross-modal, cross-cultural, cross-lingual, cross-domain, and cross-site global OER network by connecting several OER repositories. To achieve this, we started to develop a recommendation engine which inputs OER material and user activity data and outputs a list of materials that are similar to the content the user is currently viewing and matches the user's interests.

At the time of this writing we had acquired more than 80k items of OER material from four different repositories. We used this data to develop a content-based recommendation engine prototype which generates suggestions based on the material the users are accessing.

In this report we present the early prototype of recommender engine. The report includes a detailed description of the data acquired and to develop the recommender engine, a brief overview of recommender systems, and a technical description of our engine with examples.

The document is structured as follows. Section 2 focuses on the data used to generate recommendations. The data consists of both OER material information and user activity data, however at this stage we only use features from OER material. Next, in Section 3 we describe a few approaches to recommender systems and describe the engine. We also show how our engine works. Finally, we present the future steps of the recommendation engine development in Section 4 and conclude the report in Section 5.

2 OER MATERIAL AND USER ACTIVITY DATA

Before we started developing the recommendation engine we needed to understand the data that is and will be available to us. This chapter is in some parts taken from deliverable D4.1 – Initial prototype of user modelling, however it is relevant for the development of the recommender system.

We identified there are two types of data which will be useful in the development process: OER material data and user activity data. In this section we give a brief description of both types of data and how it will be used in user modelling. The data acquisition process of both types is described in D2.1 – Requirements & Architecture Report.

2.1 OER MATERIAL DATA

The OER material dataset contains information about the material available in OER repositories. It is presented in a JSON format containing the following attributes:

- **title:** the title of the material
- **provider:** the OER provider of the material
- **materialURL:** the URL to the material
- **author:** who created the material *(optional)*
- **created:** when was the material created or published *(optional)*
- **type:** what is material type, i.e. video, audio, presentation, image, text, etc.
- **language:** the language in which it is written, i.e. en, sl, es, etc.
- **metadata:** additional metadata acquired from the material, i.e. Wikipedia concepts, extracted features, etc.
- **license:** the license under which the material is shared.

The attributes labelled as optional are not necessarily present.

In the time of this writing we had OER material from 6 repositories at our disposal, out of which material from 4 repositories were pre-processed and used for the development of the recommendation engine. These repositories are:

- MIT OpenCourseWare [1]
- AMS Campus - University of Bologna [2]
- Madoc - Université de Nantes [3]
- VideoLectures.NET [4]

The remaining two OER repositories will be pre-processed in the following months. The unprocessed repositories are:

- PoliMedia – Universitat Politècnica de València [5]
- virtUOS – Unversität Osnabrück [6]

These repositories have been processed in WP3 and the result is presented in D3.1 – Learning Analytics Engine 1.0.

The processed repositories together store over 80k items of OER material. Number of materials per repository are shown in Figure 1.

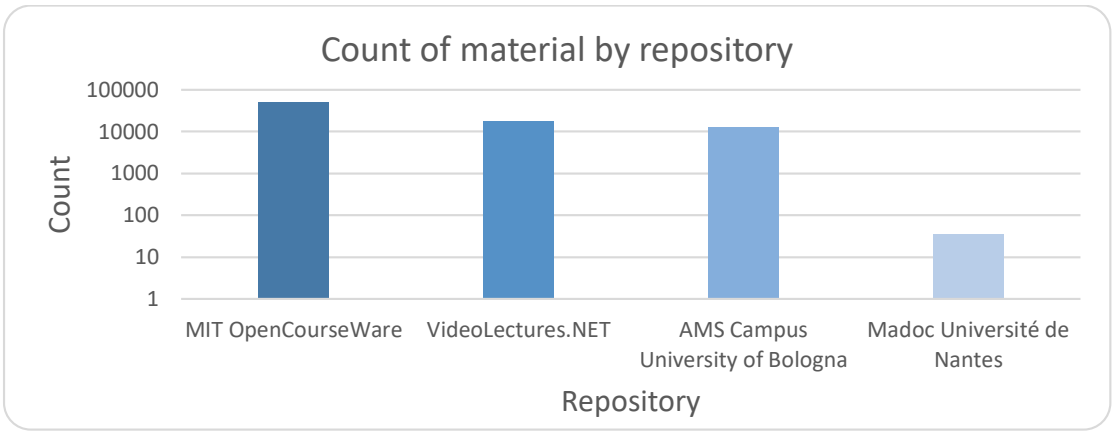

Figure 1 Number of materials per repository in logarithm scale. Most materials come from MIT OpenCouseWare followed by Videolectures.NET.

Some of the repositories offer material in different languages. All repositories together cover 103 languages, however for only 8 languages the count of available material is larger than 100. The distribution of items over languages is pictured in Figure 2. We only show languages with more than 100 items available. The "Unknown" column shows that for about 6k materials we were not able to extract the language. To acquire this information, we will improve the language extraction method in our pre-processing pipeline.

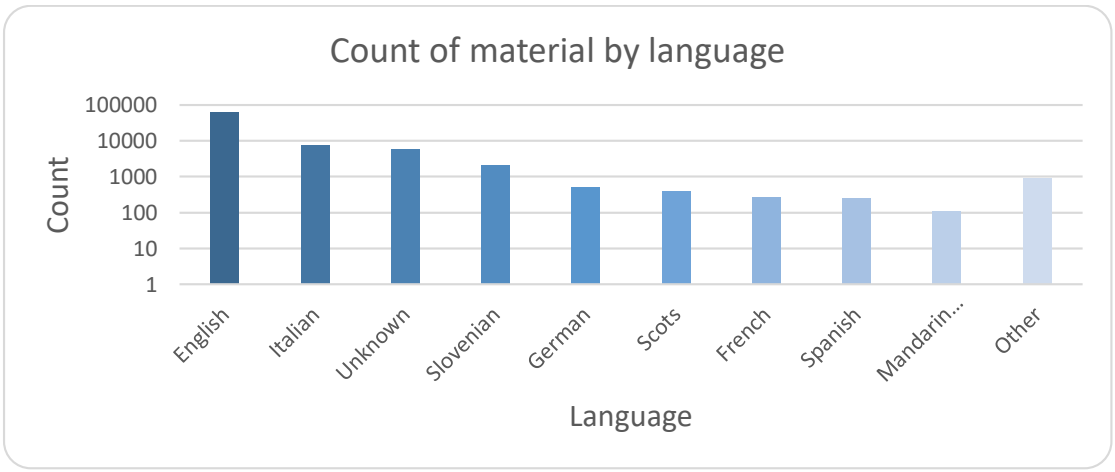

Figure 2 Number of materials per language in logarithm scale. Most of the material is in English, followed by Italian and Slovenian. Materials from which we were not able to extract language will be pre-processed again with an improved extraction method.

Repositories contain different types of material – videos, audio, and text. This results in different file types of material, meaning that the pre-processing pipeline has to handle several different file types. We visualized the distribution of materials over file types in Figure 3**Error! Reference source not found.**, however we only show types with more than 100 items available. As seen from the figure the dominant file type is text (*pdf*, *pptx* and *docx*) followed by video (*mp4*). The *msi* file type is an installer package file format used by Windows but it can also be a textual document or a presentation. If we generalize the file type distribution over all OER repositories we can conclude that the dominant file type is text – which we take into account when developing the pre-processing pipeline and recommendation engine.

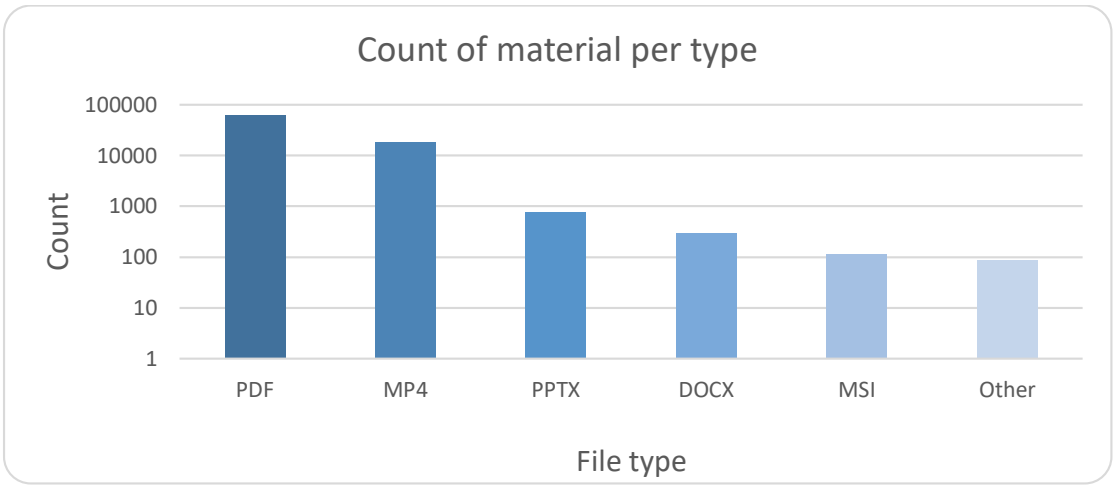

Figure 3 Number of items per file type in logarithm scale. The dominant file type is text (pdf, pptx and docx), followed by video (mp4).

Metadata is at this point the most important attribute for building the recommendation models. Metadata includes a series of values extracted from the material. One of the values is Wikipedia concepts. These represent a semantic space, i.e. vocabulary, in which we can compare materials. Additionally, Wikipedia concepts can be viewed as topics or interests the material covers.

Metadata is extracted from the material in the pre-processing pipeline which is described in D2.1 – Requirements & Architecture Report. What follows is an in-depth description of the Wikipedia concept extraction.

Due to different types of OER material we needed to include components to extract raw text from all of the provided types. We use Translectures [7] to obtain transcriptions of audio and video materials, which are then used to extract the raw text, describing its content. For textual materials we strip down the acquired document using a node.js library named textract [8]. The library inputs the document and returns the raw text of its content.

To extract the Wikipedia concepts we use Wikifier [9]. For each material we split the raw text into chunks of at most 10k characters and pass them into Wikifier which returns Wikipedia concepts found for the given chunk. These results are aggregated and stored in the materials metadata attribute.

In the future we plan to add level of difficulty to the metadata attribute. Level of difficulty, such as *primary school*, *secondary school*, or *higher education,* can be used in providing recommendations that match the users level of understanding of a certain topic. We also intend to include the findings of WP1 (Learning Rich Content Representations) and WP3 (Learning Analytics Engine) in the metadata attribute for use in the recommendation engine.

2.2 User activity data

Apart from the OER material we also acquire user activity data that describes who are the users viewing and with which technology he or she does this. We have currently acquired about 2.8M user activity records from Videolectures.NET and PoliMedia. This data will be used in the next iteration of the recommendation engine. More information about the acquired user activity data is found in D4.1 – Initial Prototype of User Modelling Architecture.

3 RECOMMENDER ENGINE

3.1 RECOMMENDER SYSTEMS OVERVIEW

Recommendation engine is a subclass of information filtering system that seeks to predict the preference a user would give to an item [10]. There are different ways of creating recommendations. The recommendation approaches we are considering are described in deliverable D4.1 – Initial Prototype of User Modelling Architecture. What follows are brief descriptions of the considered recommendation approaches.

Content based recommendation. This approach is used when user expect to get recommendations of OER material that are similar by content to the material he or she is currently viewing. This approach does not need user information since it uses material attributes such as Wikipedia concepts found under the metadata attribute. The initial prototype follows this approach and is described in the Section 3.2.

Interest based recommendation. Another approach of recommending materials is to use user's interests. As described in deliverable D4.1 we present the users interests with a set of Wikipedia concepts that were extracted from the OER materials viewed by the user and build a user model. The user model contains topics the user is interested and is used in a similar approach as content based recommendation.

Recommendation based on collaborative filtering. Collaborative filtering is a method of making automatic predictions about the interests of a user by collecting preferences or taste information from many users [11]. Here, we will expand the recommendation engine to provide the user materials which were viewed by other users with a similar preference or taste. The material will be sorted based on the number of views it got from the set of users with similar taste.

3.2 RECOMMENDER ENGINE PROTOTYPE

For the initial prototype of the recommender engine we opted for content based recommendation as such recommender systems do not suffer from the cold start problem. In this section we present how the recommender engine works and how it uses the OER material data. The material data is described in Section 1.2.1.

The basic idea of X5GON recommender engine is to provide a REST API to partner repository owners that generates cross-site, cross-modal, cross-lingual, cross-domain and cross-cultural recommendations based on the material the user is accessing on the site of that partner repository. This means that it is the OER repositories' responsibility to make API calls requesting the recommendations. The resulting recommendations are either a JSON list of recommended materials data or an automatically generated HTML with links that is ready to be embedded into OER repository site. High-level scheme is pictured in Figure 4.

The data that we need to compute the recommendations includes:

- **url:** the URL of the material the user is accessing,
- **text:** any surrounding text (search query, title of the material, abstract or description of the material, etc.), and
- **type:** [*empty*, *cosine*] – metrics used to compare Wikipedia concepts
    - empty - number of common Wikipedia concepts,
    - cosine – use similarity measure between materials raw text and the content of found Wikipedia concepts.

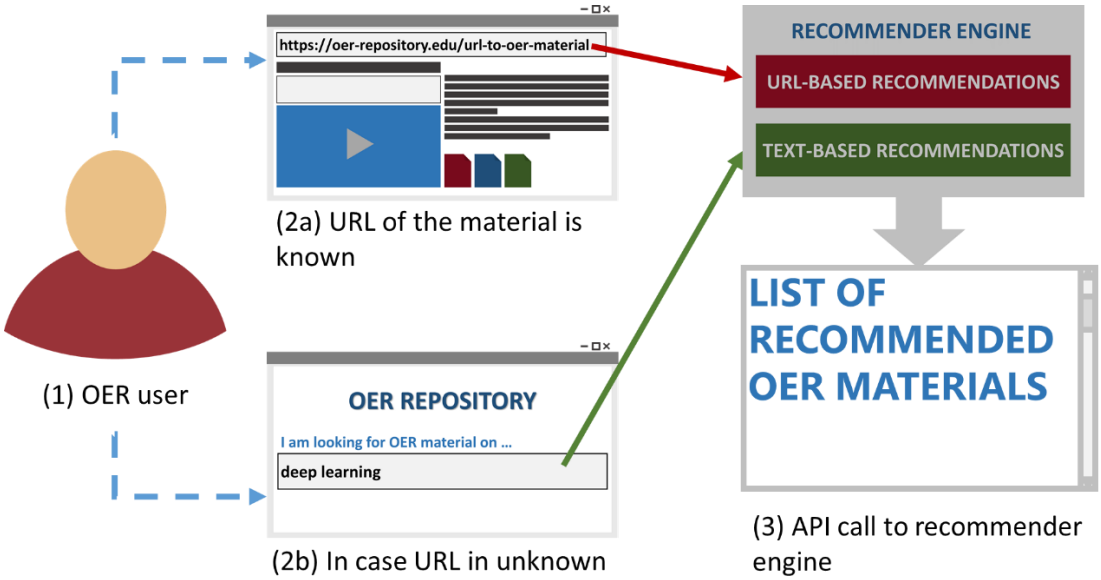

Figure 4 High-level scheme of recommender engine. The recommended materials can be acquired by either providing the material URL (2a) or by raw text (2b).

There are two separate recommender models in our recommender engine. One uses material URL to generate recommendations, whereas the other one relies on raw textual input.

URL- and text-based recommender models both use k-nearest neighbour algorithm [12] to compute the recommendations. URL-based recommender system searches for k most similar materials to the material with given URL based on the Wikipedia concepts. On the contrary text-based recommender engine we search for the materials with the most similar raw text to the query text using BOW model. Source code of the prototype can be found at [13]. The prototype considers k to be equal 100, i.e. the engine recommends 100 most similar materials based on the input.

URL-based recommendations have higher priority than text-based, meaning that if both URL and text are present in the request the engine first tries to provide the URL-based recommendations. To provide recommendations the engine must first map the provided URL to the appropriate OER material. The inability to map URL to the material in the database means that the item with the given URL is not in the database yet. This mostly happens because the material is new and has not been acquired, pre-processed and added to the database. We plan to develop a mechanism to add such material automatically when we get a request like this from a known provider.

When the URL is not provided or the engine is unable to map the URL to the material, the engine tries to provide text-based recommendations. Additionally, text-based recommendations can be used to provide suggestions from the search query from the entry site of the repository.

3.3 RECOMMENDER SYSTEMS RESULTS

As mentioned in the previous section the material recommendations can be provided in two formats. The first is by means of an automatically generated HTML with links to the recommended material. This HTML can be embedded into the OER repository using the iframe tag. What follows are examples of such generated HTML for both URL- and text-based recommendations.

Figure 5 shows recommendations in a form of HTML list with links for VideoLectures.NET video with title "*Is deep learning the new 42?"* [14]. For this demonstration we executed the following GET request in the browser: https://platform.x5gon.org/embed/recommendations?url=http://videolectures.net/kdd2016_broder_deep_learning/&type=cosine.

The results were generated by comparing Wikipedia concepts found in the materials. Additionally, the parameter *type* determines the metrics used when comparing materials using Wikipedia concepts.

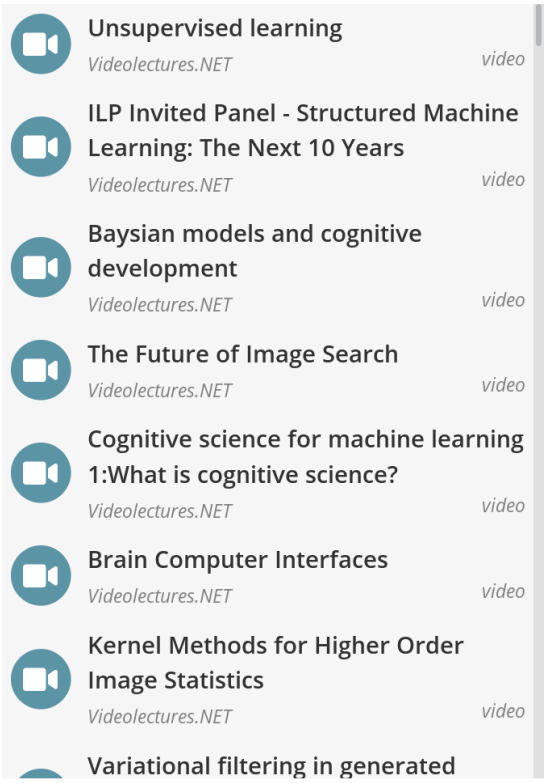

Figure 5 URL-based recommendations for video "*Is deep learning the new 42?"*

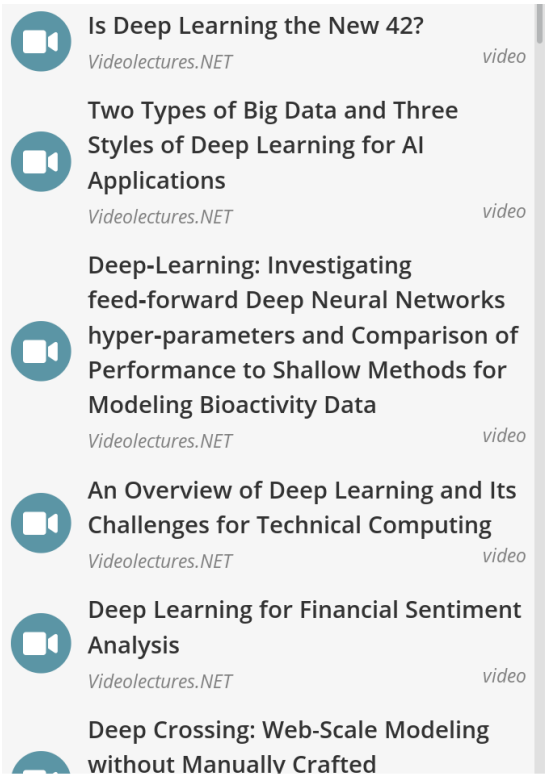

Figure 6 Text-based recommendations for text *"deep learning"*.

Similarly, Figure 6 shows recommendations for text "*deep learning"*. To get these recommendations we executed the following GET request from browser: https://platform.x5gon.org/embed/recommendations?text=deep%20learning. Here, the results are generated using the BOW model. The model compares the raw text extracted from the materials with the provided *text* value in the query. As we have seen in section 2.1, OER materials are provided in multiple languages. Because of this we use text-based recommendations as a second option – if the URL-based approach would not yield results.

Because we are using Wikipedia concepts as a semantic space representation, we are able to compare materials written in different languages. This enables us to provide cross-lingual recommendations which is one of the objectives of this work package. Additionally, because we acquire materials from different repositories and are of different types we can generate cross-site and cross-modal recommendations.

These features are already shown in the provided recommendations as shown in Figure 7. Blue frames mark cross-site and cross-modal suggestions, whereas red frame emphasizes cross-lingual case.

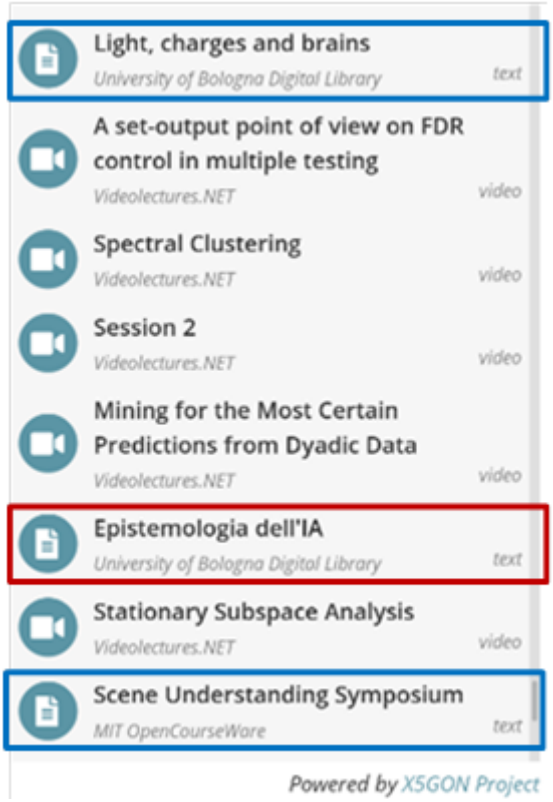

Figure 7 Cross-lingual, cross-modal, and cross-site recommendations.

The recommendation list was generated by executing the following GET request in the browser:https://platform.x5gon.org/embed/recommendations?url=http://videolectures.net/kdd2016_broder_deep_learning/. The selected items appear towards the bottom of the list.

To evaluate the recommendations we would need to get feedback from its users. To this end, we had a conversation with the Ministry of Education of Slovenia who will run one of the pilots with teachers and educators. TODO

4 FUTURE WORK

We have implemented content based recommendation method described in Section 3.2Recommender engine prototype. We set up a service on the cloud infrastructure provided by Pošta Slovenije and tested the method. This prototype uses four OER repositories as data sources. Other methods require user activity data which is provided by the X5GON user activity tracker library. We are currently in the process of collecting user activity data and expect to start developing recommendation methods from Section 3.1.

Recommendation engine currently uses content-based recommender models. The most important steps for the future include:

- adding more OER material from different sources,
- switching from purely content-based recommendations to approaches that require user activity data, such as collaborative filtering,
- including findings from WP1 and WP3, and
- testing and evaluating the recommendation engine.

5 CONCLUSION

In this report we described the early prototype of the recommendation engine.

First we made a quick overview of the data we use in the architecture. In section 2 we described the OER material data and user activity data that are used in recommender engine. The OER material data is the product of the material pre-processing pipeline within the X5GON platform described in D2.1 - Architecture and requirements report, whereas the user activity data is provided by the X5GON user activity tracker.

In Section 3 we presented the recommendation engine prototype. Firstly, we did a brief overview of common approaches to automatic generation of recommendations. Secondly, we described the current state of our recommendation engine prototype. We showed a high-level scheme and discussed two different types of recommendation models that we covered. We demonstrated how to use the engine and what results are expected with several requests.

Finally, we presented future work in Section 4 which includes adding more data, and implementing, testing, and evaluating different approaches.

REFERENCES

[1] “MIT OpenCourseWare | Free Online Course Materials,” Massachusetts Institute of Technology, [Online]. Available: https://ocw.mit.edu/index.htm. [Accessed 23 07 2018].

[2] “Benvenuto su AMS Campus - AlmaDL - Università di Bologna - AMS Campus - AlmaDL - Università di Bologna,” ALMA MATER STUDIORUM - Università di Bologna, [Online]. Available: https://campus.unibo.it/. [Accessed 23 07 2018].

[3] “Plate-forme d'Enseignement de l'Université de Nantes,” Université de Nantes, [Online]. Available: http://madoc.univ-nantes.fr/. [Accessed 23 07 2018].

[4] “Videolectures.NET - Videolectures.NET,” Videolectures.Net, [Online]. Available: http://videolectures.net/. [Accessed 23 07 2018].

[5] “media UPV,” Universitat Politècnica de València, [Online]. Available: https://media.upv.es/#/portal. [Accessed 23 07 2018].

[6] “Universität Osnabrück - Start,” Universität Osnabrück, [Online]. Available: https://www.virtuos.uni-osnabrueck.de/. [Accessed 23 07 2018].

[7] “transLectures | transcription and translation of video lectures,” translectures, [Online]. Available: http://www.translectures.eu/. [Accessed 23 07 2018].

[8] “dbashford/textract: node.js module for extracting text from html, pdf, doc, docx, xls, xlsx, csv, pptx, png, jpg, gif, rtf and more!,” [Online]. Available: https://github.com/dbashford/textract. [Accessed 23 07 2018].

[9] J. Brank, G. Leban and M. Grobelnik, “Annotating Documents with Relevant Wikipedia Concepts,” in *Proceedings of the Slovenian Conference on Data Mining and Data Warehouses (SiKDD 2017)*, Ljubljana, Slovenia, 9 October 2017.

[10] “Recommender system - Wikipedia,” Wikimedia Foundation, Inc., [Online]. Available: https://en.wikipedia.org/wiki/Recommender_system. [Accessed 23 07 2018].

[11] “Collaborative filtering - Wikipedia,” Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Collaborative_filtering. [Accessed 21 03 2018].

[12] “k-nearest neighbor algorithm - Wikipedia,” Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm. [Accessed 15 03 2018].

[13] “x5gon/src/lib/models at master - JozefStefanInstitute/x5gon,” GitHub, Inc., 2018. [Online]. Available: https://github.com/JozefStefanInstitute/x5gon/tree/master/src/lib/models. [Accessed 12 04 2018].

[14] “Is deep learning the new 42?,” VideoLectures.NET, [Online]. Available: http://videolectures.net/kdd2016_broder_deep_learning/. [Accessed 18 July 2018].