Preparing multi-modal data for natural language processing

Erik Novak
Jožef Stefan Institute
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia
erik.novak@ijs.si

Jasna Urbančič
Jožef Stefan Institute
Ljubljana, Slovenia
jasna.urbancic@ijs.si

Miha Jenko
Jožef Stefan Institute
Ljubljana, Slovenia
miha.jenko@ijs.si

ABSTRACT

In education we can find millions of video, audio and text educational materials in different formats and languages. This variety and multimodality can impose difficulty on both students and teachers since it is hard to find the right materials that match their learning preferences. This paper presents an approach for retrieving and recommending items of different modalities. The main focus is on the retrieving and preprocessing pipeline, while the recommendation engine is based on the k-nearest neighbor method. We focus on educational materials, which can be text, audio or video, but the proposed procedure can be generalized on any type of multi-modal data.

KEYWORDS

Multi-modal data preprocessing, machine learning, feature extraction, recommender system, open educational resources

ACM Reference Format:
Erik Novak, Jasna Urbančič, and Miha Jenko. 2018. Preparing multi-modal data for natural language processing. In *Proceedings of Slovenian KDD Conference (SiKDD'18).* ACM, New York, NY, USA, Article 4, 4 pages. https://doi.org/10.475/123_4

1 INTRODUCTION

There are millions of educational materials that are found in different formats – courses, video lectures, podcasts, simple text documents, etc. Because of its vast variety and multimodality it is difficult for both students and teachers to find the right materials that will match their learning preferences. Some like to read a short scientific papers while others just like to sit back and watch a lecture that can last for hours. Additionally, materials are written in different languages, which is a barrier for people who are not fluent in the language the material is written in. Finding a good approach of providing educational material would help improving their learning experience.

In this paper we present a preprocessing pipeline which is able to process multi-modal data and input it in a common semantic space. The semantic space is based on Wikipedia concepts extracted from the content of the materials. Additionally, we developed a content based recommendation model which uses Wikipedia concepts to find similar items based on the model input. Throughout the paper we focus on educational material but the approach can be generalized to other multi-modal data sets.

The reminder of the paper is structured as follows. In section 2 we go over related work. Next, we present the data preprocessing pipeline which is able to process different types of data – text, video and audio – and describe each component of the pipeline in section 3. A content based recommendation model that uses Wikipedia concepts to compare materials is presented in section 4. Finally, we present future work and conclude the paper in section 5.

2 RELATED WORK

In this section we present the related work which the rest of the paper is based on. We split this section into subsections – multi-modal data preprocessing and recommendation models.

Multi-modal Data Preprocessing. Multi-modal data can be seen as classes of different data types from which we can extract similar features. In the case of educational material the classes are video, audio and text. One of the approaches is to extract text from all class types. In [6] the authors describe a Machine Learning and Language Processing automatic speech recognition system that can convert audio to text in the form of transcripts. The system can also process video files as they are also able to extract audio from it. Their model was able to achieve a 13.3% word error rate on an English test set. These kind of systems are useful for extracting text from audio and video but would need to have a model for each language.

Recommendation models. These models are broadly used in many fields – from recommending videos based on what the user viewed in the past, to providing news articles that the user might be interested in. One of the most used approaches is based on collaborative filtering [16], which finds users that have similar preferences with the target user and recommends items based on their ratings. Recommender systems now do not contain only one algorithm but multiple which return different recommendations.

Authors of [10] discuss about the various algorithms that are used in the Netflix recommender system (top-n video ranker, trending now, continue watching, and video-video similarity), as well as the methods they use to evaluate their system. A high level description of the Youtube recommender system is found in [3]. They developed a candidate generation model and a ranking model using deep learning. Both Netflix and Youtube recommend videos based on users' interaction with them and the users history. To some extent this can be used for educational resources but cannot be generalized on the whole multi-modal data set since we cannot acquire data about users' interaction with, for instance, text.

SiKDD'18, October 2018, Ljubljana, Slovenia

ACM ISBN 123-4567-24-567/08/06.
https://doi.org/10.475/123_4

A collaborative filtering based recommendation system for the educational sector is presented in [8]. They evaluated educational content using big data analysis techniques and recommended courses to students by using their grades obtained in other subjects. This gives us insight into how recommendations can be used in education but our focus is to recommend educational materials rather than courses. In a sense courses can be viewed as bundles of educational material; thus, our interest is recommending "parts of courses" to the user.

3 DATA PREPROCESSING

In this paper we focus on open educational resources (OER), which are freely accessible, openly licensed text, media, and other digital assets that are useful for teaching, learning and assessing [21]. These are found in different OER repositories maintained by universities, such as MIT OpenCourseWare [12], Università di Bologna [7], Université de Nantes [4] and Universitat Politècnica de València [5], as well as independent repositories such as Videolectures.NET [20], a United Nations award-winning free and open access educational video lectures repository.

For processing the different OER we developed a preprocessing pipeline that can handle each resource type and output metadata used for comparing text, audio and video materials. The pipeline is an extension of the one described in [11]; its architecture is shown in figure 1. What follows are the descriptions of each component in the preprocessing pipeline.

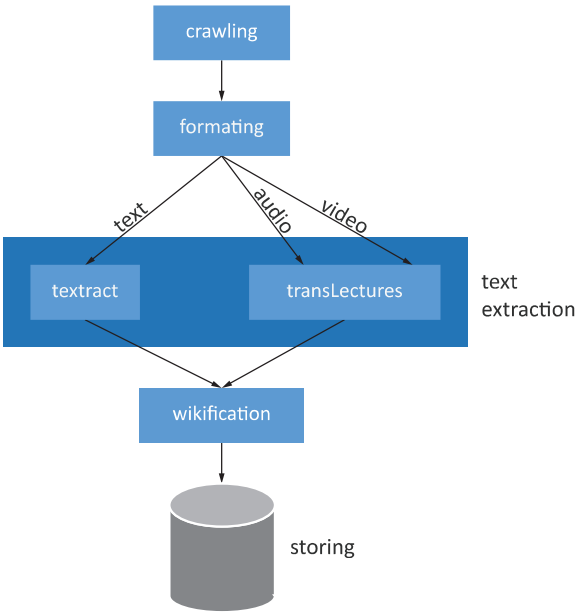

Figure 1: The preprocessing pipeline architecture. It is designed to handle each data type as well as extract features to support multi- and cross-linguality.

Crawling. The first step is to acquire the educational materials. We have targeted four different OER repositories (MIT OpenCourseWare, Università di Bologna, Université de Nantes and Videolectures.NET), for which we used their designated APIs or developed custom crawlers to acquire their resources. For each material we acquired its metadata, such as the materials title, url, type, language in which it is written and its provider. These values are used in the following steps of the pipeline as well as to represent the material in the recommendations.

Formatting. Next, we format the acquired material metadata. We designate which attributes every material needs to have as well as set placeholders for the features extracted in the following steps of the pipeline. By formatting the data we set a schema which makes checking which attributes are missing easy. We do not have a mechanism for handling missing attributes in the current pipeline iteration but we will dedicate time to solve this problem in the future.

Text Extraction. The third step, we extract the content of each material in text form. Since the material can be a text, video or audio file to handled each file type separately.

For text we employed *textract* [1] to extract raw text from the given text documents. The module omits figures and returns the content as text. The extracted text is not perfect - in the case of materials for mathematics it does not know how to represent mathematical equations and symbols. In that case, it replaces the equations with textual noise. Currently we do nothing to handle this problem and use the output as is.

For video and audio we use the subtitles and/or transcriptions to represent the materials content. To do this, we use *transLectures* [18] which generates transcriptions and translations of a given video and audio. The languages it supports are English, Spanish, German and Slovene. The output of the service is in dfxp format [17], a standard for xml caption and subtitles based on timed text markup language, from which we extract the raw text.

Wikification. Next, we send the material through wikification - a process which identifies and links material textual components to the corresponding Wikipedia pages [15]. This is done using Wikifier [2], which returns a list of Wikipedia concepts that are most likely related to the textual input. The web service also supports cross- and multi-linguality which enables extracting and annotating materials in different languages.

Wikifier's input text is limited to 20k characters, because of which longer text cannot be processed as a whole. We split longer text into chunks of at most 10k characters and pass them to Wikifier. Here we are careful not to split the text in the middle of a sentence and if that is not possible, to at least not split any words.

We split the text as follows. First we make a 10k characters long substring of the text. Next, we identify the last character in the substring that signifies the end of a sentence (a period, a question mark, or an exclamation point) and split it at that character. If there is no such character we find the last whitespace in the substring and split it there. In the extreme case where no whitespaces are found we take the substring as is. The substring becomes one chunk of the original text. We repeat the process on the remaining text until it is fully split into chunks.

When we pass these chunks into Wikifier, it returns Wikipedia concepts related to the given chunk. These concepts also contains

the Cosine similarity between the Wikipedia concept page and the given input text. To calculate the similarity between the concept and the whole material we aggregated the concepts by calculating the weighted sum

$$S_k = \sum_{i=1}^{n} \frac{L_i}{L} s_{ki},$$

where S_k is the aggregated Cosine similarity of concept k, n is the number of chunks for which Wikifier returned concept k, L_i is the length of chunk i, L is the length of the materials raw text, and s_{ki} is the Cosine similarity of concept k to chunk i. The weight $\frac{L_i}{L}$ represents the presence of concept k, found in chunk i, in the whole material. The aggregated Wikipedia concepts are stored in the materials metadata attribute.

Data Set Statistics. In the final step, we validate the material attributes and store it in a database. The OER material data set consists of approximately 90k items. The distribution of materials over the four repositories is shown in figure 2.

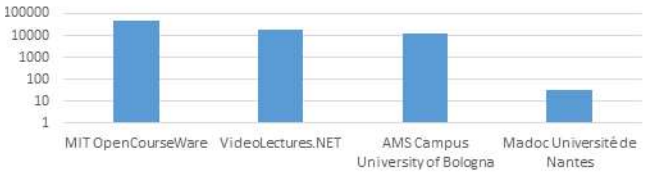

Figure 2: Number of materials per repository crawled in logarithm scale. Most materials come from MIT OpenCourseWare followed by Videolectures.NET.

Some of the repositories offer material in different languages. All repositories together cover 103 languages, however for only 8 languages the count of available materials is larger than 100. The distribution of items over languages is shown in figure 3 where we only show languages with more than 100 items available. Most of the materials is in English, followed by Italian and Slovene. 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 preprocessing pipeline.

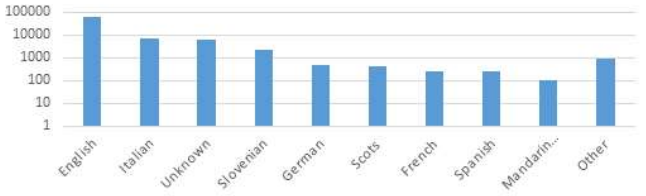

Figure 3: Number of materials per language in logarithm scale. Most of the material is in English, followed by Italian and Slovenian.

As shown in before the preprocessing pipeline is designed to handle different types of material - text, video and audio. Each type can be represented in various file formats, such as pdf and docx for text, wmv and mp4 for video, and mp3 for audio. We visualized the distribution of materials over file types in figure 4, but we only show types with more than 100 items available.

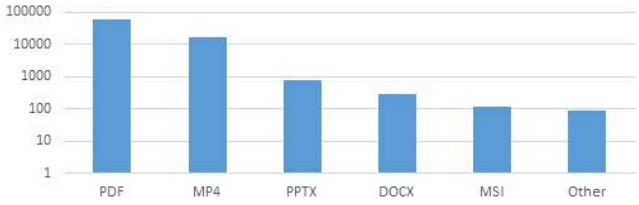

Figure 4: Number of items per file type in logarithm scale. The dominant file type is text (pdf, pptx and docx), followed by video (mp4).

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. This will be taken into count when improving the preprocessing pipeline and recommendation engine.

4 RECOMMENDER ENGINE

There are different ways of creating recommendations. Some employ users' interests while other are based on collaborative filtering. In this section we present our content based recommendation engine which uses the k-nearest neighbor algorithm [13]. What follows are descriptions of how the model generates recommendations based on the user's input, which can be either the identifier of the OER in the database or a query text.

Material identifier. When the engine receives the material identifier (in our case the url of the material) we first check if the material is in our database. If present, we search for k most similar materials to the one with the given identifier based on the Wikipedia concepts. Each material is represented by a vector of its Wikipedia concepts where each value is the aggregated Cosine similarity of the corresponding Wikipedia concept page to the material. By calculating the Cosine similarity between the materials the engine then selects k materials with the highest similarity score and returns them to the user. Because of the nature of Wikipedia concepts this approach returns materials written in different languages - which helps overcoming the language barrier.

Query text. When the engine receives the query text we search for materials with the most similar raw text using the bag-of-words model. Each material is represented as a bag-of-words vector where each value of the vector is the tf-idf of the corresponding word. The materials are then compared using the Cosine similarity and the engine again returns the k materials that have the highest similarity score. This approach is simple but it is unable to handle multilingual documents. This might be overcome by first sending the query text to Wikifier to get its associated Wikipedia concepts and use them in a similar way as described in the *Material identifier* approach.

4.1 Recommendation Results

The described recommender engine is developed using the QMiner platform [9] and is available at [14]. When the user inputs a text query the system returns recommendations similar to the given text. These are shown as a list where each item contains the title, url, description, provider, language and type of the material. Clicking on an item redirects the user to the selected OER.

We have also discussed with different OER repository owners and found that they would be interested in having the recommendations in their portal. To this end, we have developed a compact recommendation list which can be embedded in a website. The recommendations are generated by providing the material identifier or raw text as query parameters in the embedding url. Figure 5 shows the embed-ready recommendation list.

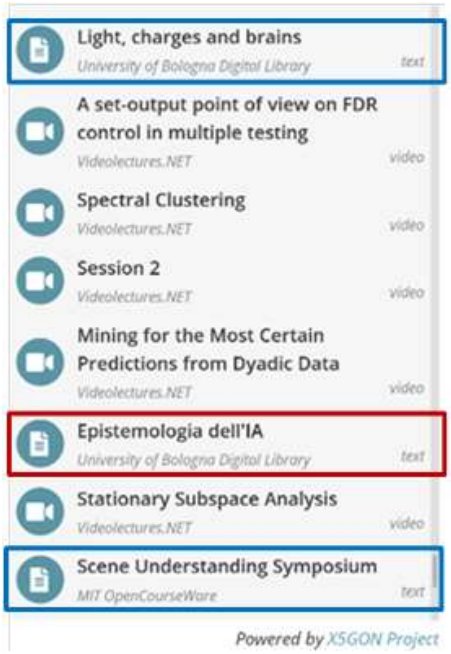

Figure 5: An example of recommended materials for the lecture with the title "Is Deep Learning the New 42?" published on Videolectures.NET [19]. The figure shows cross-lingual, cross-modal, and cross-site recommendations.

The recommendation list consists of the top 100 materials based on the query input. As shown in the figure the recommendation contain materials of different types, are provided by different repositories and written in different languages. We have not yet evaluated the recommendation engine but we intend to do it in the future.

5 FUTURE WORK AND CONCLUSION

In this paper we present the methodology for processing multimodal items and creating a semantic space in which we can compare these items. We acquired a moderately large open educational resources data set, created a semantic space with the use of Wikipedia concepts and developed a basic content based recommendation engine.

In the future we will evaluate the current recommendation engine and use it to compare it with other state-of-the-art. We intend to use A/B testing to optimize the models based on the user's interaction with them. We wish to improve the engine by collecting user activity data to determine what materials are liked by the users, explore different deep learning methods to improve results, and develop new representations and embeddings of the materials.

We also aim to improve the preprocessing pipeline by improving text extraction methods, handle missing material attributes, and adding new feature extraction methods to determine the topic and scientific field of the educational material as well as their quality.

ACKNOWLEDGMENTS

This work was supported by the Slovenian Research Agency and X5GON European Unions Horizon 2020 project under grant agreement No 761758.

REFERENCES

[1] David Bashford. 2018. GitHub - dbashford/textract: node.js module for extracting text from html, pdf, doc, docx, xls, xlsx, csv, pptx, png, jpg, gif, rtf and more! https://github.com/dbashford/textract. Accessed: 2018-09-03.
[2] Janez Brank, Gregor Leban, and Marko Grobelnik. 2017. Annotating documents with relevant Wikipedia concepts. *Proceedings of SiKDD.*
[3] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems.* ACM, 191–198.
[4] Université de Nantes. 2018. Plate-forme d'Enseignement de l'Université de Nantes. http://madoc.univ-nantes.fr/. Accessed: 2018-09-03.
[5] Universitat Politècnica de València. 2016. media UPV. https://media.upv.es/#/portal. Accessed: 2018-09-03.
[6] Miguel Ángel del Agua, Adrià Martínez-Villaronga, Santiago Piqueras, Adrià Giménez, Alberto Sanchis, Jorge Civera, and Alfons Juan. 2015. The MLLP ASR Systems for IWSLT 2015. In *Proc. of 12th Intl. Workshop on Spoken Language Translation (IWSLT 2015).* Da Nang (Vietnam), 39–44. http://workshop2015.iwslt.org/64.php
[7] Università di Bologna. 2018. Universita di Bologna. https://www.unibo.it/it. Accessed: 2018-09-03.
[8] Surabhi Dwivedi and VS Kumari Roshni. 2017. Recommender system for big data in education. In *E-Learning & E-Learning Technologies (ELELTECH), 2017 5th National Conference on.* IEEE, 1–4.
[9] Blaz Fortuna, J Rupnik, J Brank, C Fortuna, V Jovanoski, M Karlovcec, B Kazic, K Kenda, G Leban, A Muhic, et al. 2014. » QMiner: Data Analytics Platform for Processing Streams of Structured and Unstructured Data «, Software Engineering for Machine Learning Workshop. In *Neural Information Processing Systems.*
[10] Carlos A Gomez-Uribe and Neil Hunt. 2016. The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems (TMIS)* 6, 4 (2016), 13.
[11] Erik Novak and Inna Novalija. 2017. Connecting Professional Skill Demand with Supply. *Proceedings of SiKDD.*
[12] Massachusetts Institute of Technology. 2018. MIT OpenCourseWare | Free Online Course Materials. https://ocw.mit.edu/index.htm. Accessed: 2018-09-03.
[13] Leif E Peterson. 2009. K-nearest neighbor. *Scholarpedia* 4, 2 (2009), 1883.
[14] X5GON Project. 2018. X5GON Platform. https://platform.x5gon.org/search. Accessed: 2018-09-04.
[15] Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and global algorithms for disambiguation to wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1.* Association for Computational Linguistics, 1375–1384.
[16] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web.* ACM, 285–295.
[17] Speechpad. 2018. DFXP (Distribution Format Exchange Profile) | Speechpad. https://www.speechpad.com/captions/dfxp. Accessed: 2018-09-04.
[18] transLectures. 2018. transLectures | transcription and translation of video lectures. http://www.translectures.eu/. Accessed: 2018-09-03.
[19] VideoLectures.NET. 2018. Is Deep Learning the New 42? - Videolectures.NET. http://videolectures.net/kdd2016_broder_deep_learning/. Accessed: 2018-09-03.
[20] VideoLectures.NET. 2018. VideoLectures.NET - VideoLectures.NET. http://videolectures.net/. Accessed: 2018-09-03.
[21] Wikipedia. 2018. Open educational resources - Wikipedia. https://en.wikipedia.org/wiki/Open_educational_resources. Accessed: 2018-09-03.