

# A Web-based Recommendation Mechanism for Learning Objects Combining Ontologies and Zero-shot Learning

Dimitrios K. Charalampopoulos and Dimitrios A. Koutsomitropoulos

Computer Engineering and Informatics Dpt., University of Patras, 26500 Patras, Greece  
{charalamp, koutsomi}@ceid.upatras.gr

**Abstract.** Content management, selection and reuse is a difficult task especially for educational purposes. Flat metadata aggregations lack semantic interoperability and content is often poorly indexed thus making its discovery and integration troublesome. In this paper we propose a web interface for managing learning objects that leverages ontologies, automated subject classification and scoring to aid users and curators overcome these obstacles. We examine and document requirements and design decisions spanning data handling, storage, display, and user interaction. Through research and comparative evaluation, we choose an appropriate database layer and corresponding web framework for the application’s development. The app is capable of consuming metadata from other repositories but also maintains a local pool of learning object metadata where users can collaboratively contribute on content indexing and recommendation.

**Keywords:** Ontologies, Learning Objects, Metadata, JavaScript Frameworks.

## 1 Introduction

Vast data availability can make it challenging for someone to store, search and find the exact thing one is looking for [16]. For this reason, digital libraries containing nearly every type of data have been created. More importantly, educational information is stored in chunks of data, so it can be easily accessed by everyone. These chunks of data are called Learning Objects (LOs) and they contain metadata for the educational subject they are addressing [7]. As rich as the object’s metadata may be, there is however a considerable semantic gap between annotations originating from different sources. Consequently, content searching, selection and further indexing is hampered and often requires manual resolution.

Building on previous work [12][11] in this paper we propose a web-based front-end for a learning object repository based on dynamic web frameworks using JavaScript (JS) and Node.js. We primarily focus on creating a web interface that would ideally be implemented for use in all educational levels [3]. First, by leveraging the notion of a Learning Object Ontology Repository (LOOR), we have shown that it is possible to search through various educational repositories’ metadata and amalgamate their semantics into a common LO ontology [12]. Then we commenced with automatic subject classification of LOs using keyword expansion and referencing standard taxonomic vocabularies for thematic classification [11]. The proposed front-end takes advantage of the LOOR and gives users the ability to display, manage and enrich LOs. In addition, it adds a persistency layer that allows for content already sought for and

indexed by curators to be available for others to reuse. Finally, it integrates with a novel validation service that checks semantic similarity between subject tags proposed by the LOOR and the LO itself and produces recommendations that can help users in selecting content and deciding on the most appropriate keywords. Source code and a demo of our implementation are available at: <https://github.com/Dcharalamp/React-LOR>.

The main contributions of the work presented in this paper are summarized as follows:

- Comparative assessment of the available JS frameworks and selection based on our needs.
- Design and implementation of a LOOR web interface based on the chosen framework that provides for improved and collaborative management, selection and indexing of LOs.
- Development of a management system ideal for LO integration. This will help with discovery, reusability, and enhancement of each LO.
- A novel recommendation and scoring mechanism for subject tags that is based on zero-shot machine learning [20] and is therefore light on resources.

To give a more thorough understanding of what follows, we first analyze the base concepts and review related work in the field (Section 2). Next, in Section 3 we present the design and architecture of our application. We compare JavaScript web frameworks to identify what is the best suited one for our purposes. After that, we discuss a series of requirements our front-end to the LOOR may fulfill, as well as the reasoning for their implementation. In Section 4, we examine some indicative use cases that exemplify the functionality of our approach. Finally, in the last section we discuss our conclusions and future work.

## 2 Background and Related work

An LO is a collection of combined information and data that aim to be used for educational purposes [7]. Most of the time, LOs do not stand on their own, but they are well organized in big collections inside repositories. A popular standard for describing LOs is the IEEE Learning Object Metadata (LOM) model. The purpose of LOM is to support the reusability of learning objects, to aid discoverability as well as to facilitate their interoperability, search, evaluation and acquisition [9].

Based on LOM we identify a least common set of elements available on various educational repositories, including MERLOT II, a large archive of LOs [14], Europe PubMed Central, a major repository of biomedical literature [5], ARIADNE finder, a European infrastructure for accessing and sharing learning resources [18] and openarchives.gr, the entry point for Greek scholarly content. This set includes LOM properties such as `lom:title`, `lom:description`, `lom:keyword` and `lom:identifier` and forms the basis for a *Learning Object Ontology Schema* to be used for immediate ingestion of learning objects into the LOOR [12]. This schema serves the metadata “semantification” process, i.e., the transformation of the textual information captured by a metadata instance into a semantically enriched and thus machine-understandable format [8]. For the LOM-specific entities, the official LOM namespace has been used (<http://ltsc.ieee.org/xsd/LOM/>, prefix `lom:`).

Similar to our approach, many researchers have long proposed the use of an ontology

for describing both the content and structure of LOs, as well as of an ontology for modelling LO categories [6]. In addition, earlier research focuses on comparative studies involving different metadata standards [15]. The authors are addressing the need of metadata annotations for efficient retrieval of learning materials within learning object repositories. They also identify the advantages of annotating the documents with some standard metadata schemas for making the former reusable and interoperable between different learning systems.

Next, we review some LO repositories that are oriented towards educational purposes, so as to identify common ground as well as differences to our approach. The first example we will discuss is the MIT repository for open courses, called MIT OpenCourseWare [4]. If someone visits this repository one can clearly see that there is a search option based on the title of the LO. As far as we know, there are no other metadata involved in the search operation, nor any advanced search option. A typical pagination of the results can be seen but with no filtering or sorting option. Last but not least, there is no option to alter the LOs in any way or add/delete LOs and also there is no ability to search for LOs from an external source.

We also consider a more advanced repository, the Multimedia Educational Resource for Learning and Online Teaching also known as MERLOT. It provides access to curated learning tools and subjects and it is addressed mostly to professors and students of the highest educational level [14]. In this case, the search function takes into consideration way more metadata fields than before. An interesting feature is the scoring of the LOs by the users themselves. This provides additional information for the quality of the given object. Regarding the ability to alter LOs and their metadata, the repository states that signed users are permitted to upload new LOs but there is no option to delete or edit existing ones. Finally, fetched results are both from the repository itself and external sources, like other libraries or the Web.

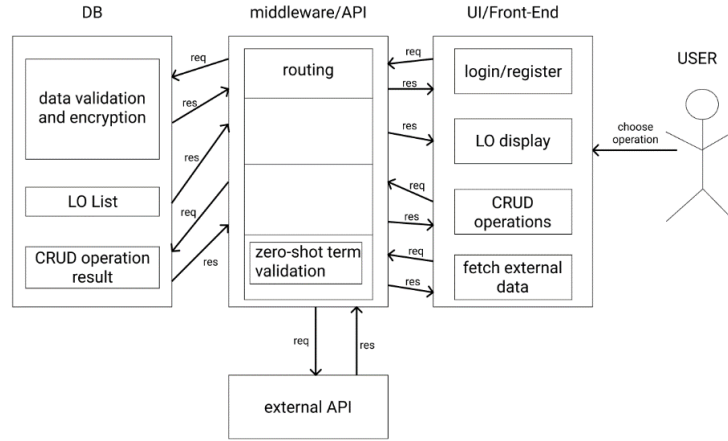
From the discussion above, we get a clear view of what features need to be present and what new features will be desirable additions. Our approach is more comparable with the 2<sup>nd</sup> example we discussed. The web interface is going to be used mostly by students; therefore, a variety of options and pagination features is more suitable than a restricted approach. We also like the idea of fetching objects from an external source, but with some differences. We would like to give our users the ability to add external resources to our own repository. This way, the intellectual wealth of the LOOR will increase and users will be able to reuse the LOs more easily. If we follow this approach however, a way to handle the duplicate ontology entries that may occur is advised.

In earlier work we have investigated and documented the positive effects of query expansion when harvesting Open Educational Resources (OERs) and for subject classification for example, boosting recall of retrieval by a factor of 4-8 [11]. Initial search keywords are reused to discover matching and related terms within domain knowledge thesauri, such as the Medical Subject Headings (MeSH) [19]. These matches are injected as semantic subject annotations into selected LOs, using the `lom:keyword` property of the LO Ontology Schema. To assess the quality of discovered subject tags, we tap into deep language models based on word embeddings, including BART [1] and XLM-RoBERTa (XLM-R) [20] and acquire validation scores for each tag that can be displayed and stored through the proposed application. To keep low on training resources we employ the idea of zero-shot classification, that is, to assign an appropriate label to a piece of text, irrespective of the text domain and the aspect

described by the label. Effectively, zero-shot classification attempts to classify text even in the absence of labeled data.

### 3 Design and Architecture

In this section we describe the design process we followed for creating the web interface. First, we present a brief workflow of the interface’s architecture.



**Fig. 1.** Brief UML Case Diagram of the application.

As can be seen from Figure 1, our architecture consists of 3 main axes: The database, where all information about user login and registration is stored, as well as every LO’s metadata and its corresponding CRUD (Create, Remove, Update, Delete) operations; the middleware, which consists of API routes that direct the requests and the responses of each operation; and finally, the front-end from where the user can interact with the features of the interface. We selected MongoDB as our database because, thanks to its document-oriented nature, it can handle this type of data efficiently and can also cooperate with any JS framework we choose. What every route has in common, is that they are directly associated with an API call. We will focus on each step and each route of the workflow later in this section.

#### 3.1 JavaScript Web Frameworks

Before proceeding with the implementation, an important matter that should be taken into consideration is the selection of an appropriate web framework for developing such a web application and the criteria which this choice should be based upon. Undeniably, Because of the proliferation of current web frameworks, we decided to focus on the most popular ones, by taking into consideration criteria such as the number of stars in GitHub, google trends, stability etc. [2]. The 3 most popular JS frameworks appear to be Angular, React and Vue [2][17][13].

### **Angular**

It is a Typescript focused framework that follows the Model-View-Controller (MVC) architecture. Angular is a complete framework. It does not require additional libraries or tools to create a feature. This at first glance sounds like an important advantage, but in reality, can lead to problems. The huge space of its library can cause performance problems and slow apps, especially on simple interfaces with 2-3 ordinary features. Scheduled updates for the framework are taking place every 6 months. Most of the time, these updates introduce important changes that alter the behavior of current working apps [2]. Therefore, we cannot rely on this framework for stability in our apps, especially in the long run.

### **React**

It is an open-source framework designed by Facebook. It does not follow the MVC architecture; rather, it is based on single-direction data flow which can provide greater control during a project. It is especially suited for creating single-page web applications. The downside of this framework is that it is not complete. Because it is a mostly UI-focused framework, certain features are not available and use of third-party libraries is mandatory. React's library size is small but tends to grow in size based on the needs of the application. The creators have stated that the stability of this framework is highly important for them, because large companies use this framework on a big scale [17].

### **Vue**

It is an open-source framework for creating single-page interfaces and it is the newest amongst them. The big advantage this framework has is its size. With a library size of 20KB, it can offer high speeds in software development. Evidently, a lot of plug-ins are missing from the main core of its library but, like React, additions can be made. Vue is not supported or used by large companies and its community is entirely open-source driven [13]. Therefore, although its creators have stated that updates will not be a problem in the future, we cannot be entirely sure about the stability of our apps, especially in the long run.

Since we aim to create an application that is going to be used for educational purposes, stability is very important. Another thing we must consider is the target-group this application aims for. The majority are going to be students, so having a very strong UI-focused framework that handles single-page interfaces better than others will certainly help. In terms of library size, our application is not going to be too feature-heavy, therefore a framework that easily allows to plug/unplug desired tools and libraries will be the better choice. Based on all the above, we decided that the best JS framework to use for these purposes is React.js.

## **3.2 Design Requirements**

Most features should be oriented towards the application's target-group, that is, students and instructors. Not all features will be available to everyone: users will be separated in 2 different levels of accessibility. Simply put, faculty and curators will have full control of CRUD operations upon LOs, where students will mostly have read-only rights. Finally, we need to implement a feature where external repositories can contribute in some way to the local repository. We also need to find a solution about possible duplicates that might occur. A different coloring scheme seems to be a method of handling such problems.

**LO search and recommendation.** Apart from CRUD operations, users should have the ability to search for specific LOs based on keywords. These keywords are searched against specific metadata fields as they are expressed within the LO Ontology Schema. Given the least common subset of elements (Section 2), our approach is to use the ontology properties `lom:title`, `lom:keyword` and `lom:description` for faceted search. The same approach will be followed when fetching LOs from an external online repository. In this case, however, we have to take into consideration possible duplicate (or semi-duplicate) results that might occur during this fetch. Duplicates may occur when metadata fetched from external repositories are already stored locally but may contain different subject assignments. For this reason, we plan to create a recommendation system that informs the user about the metadata that each repository suggests. For example, let's say that a LO in our repository has the subject keywords "viruses" and "defective". The same LO gets fetched from an external repository but this time it has "viruses" and "vaccine" as its subject keywords. The metadata record will not be replicated twice; instead, our interface will inform the user which repository recommends which keywords with different coloring for a better and more user-friendly experience.

**Persistent storage and reusability.** Reusability of the LOs is highly important for the design procedure. We would like to provide direct ways in which the LOs are being displayed so reusability will be straightforward task. New LOs should be added manually but results from external repositories should be also accommodated. For this reason, a user can search an external repository, select some of the results and store them to the database. The metadata fields completed manually or fetched from another repository populate the LO Ontology Schema and are stored within the local database. That way, the LO and its annotations are being replicated locally and reused easier, faster and following a more direct approach. On top of that, we want to combine this feature with a strategy to export either the LOs themselves or the entire way the data is being saved and managed. To this end, we can take advantage of REST APIs and facilitate metadata export for interoperability purposes. We have created multiple routings, each covering a different option: if a user wants to export the entire DB, with the right API call, all data is displayed to the user in the same shape and form that are stored inside the DB (JSON format). On the other hand, when the user wants to export specific LOs based on their unique id, this can also be achieved by calling the appropriate API URL.

**Semantic indexing and zero-shot learning.** Another important feature is the ability to select and/or edit the subject annotations of available learning resources. Typically, a curator would be responsible for entering appropriate subject tags, possibly originating from a controlled vocabulary, a process that would streamline further discovery and reuse by other users. To this end, the LOOR provides a set of automatic subject keywords through the semantic matching process described in Section 2. Further, each keyword is coupled with a validity score produced by zero-shot classification (zero-shot term validation in Figure 1). To fetch scores, we leverage the Huggingface model hub Inference API [10]. Out of the models available on the hub that implement zero-shot inference we have chosen *bart-large-mnli* and *xlm-roberta-large-xnli*, which exhibits, among others, multilingual capabilities. Taking advantage of these suggestions and accompanying scores, a user, without having any previous knowledge

upon the subject of an item, can classify the object itself through the recommended tags and decide on which they should keep and which to discard.

## 4 Functionality and Use Cases

In this section, we present a series of use cases and corresponding examples for the LOOR and the LO ontology. For the purposes of this, we use a specific LOOR implementation<sup>1</sup> that fetches data from the various data sources mentioned in Section 3.2 and uses, among others, the MeSH taxonomy for the automated subject annotations (keywords). Some trivial but nevertheless important use cases are not going to be described in detail, such as login/register, pagination, sorting, and all sorts of quality-of-life features as well as the initialization of the application. We will focus on the users that have access to CRUD operations, as this supersedes use cases for which users have read-only rights.

### 4.1 CRUD operations

**Adding a Learning Object.** There are multiple ways a user can add a new LO to the repository. The first, most common method is by filling a form. As shown in Fig.2, another way one can add objects to the database is via file upload. This way the user can concurrently add many LOs. After the file is uploaded, some validation checks occur (file needs to have the correct extension and objects inside it need to have proper form) and duplicates are filtered out. The progress bar on top of the form will inform the user when the upload is completed. The final option is to add an object that was fetched from an external online repository. This is another use case (*fetching data from external online repository*) and is explained in section 4.2.

The screenshot shows a web application interface for 'Learning Objects Digital Library'. At the top, there is a navigation bar with 'About', 'test1@gmail.com', and 'Logout' links. The main content area features a 'Register Object' form. The form has a title bar with a progress indicator and an 'Upload' button. Below the title bar, there are five input fields: 'ID', 'Title', 'Description', 'Identifier', and 'Keywords'. The 'Keywords' field has a placeholder text: 'For more than 1 keyword, separate with comma'. At the bottom of the form is a 'Register' button. The background of the page is a blurred image of a library with bookshelves and a laptop.

**Fig. 2.** Adding an LO's metadata to the local repository.

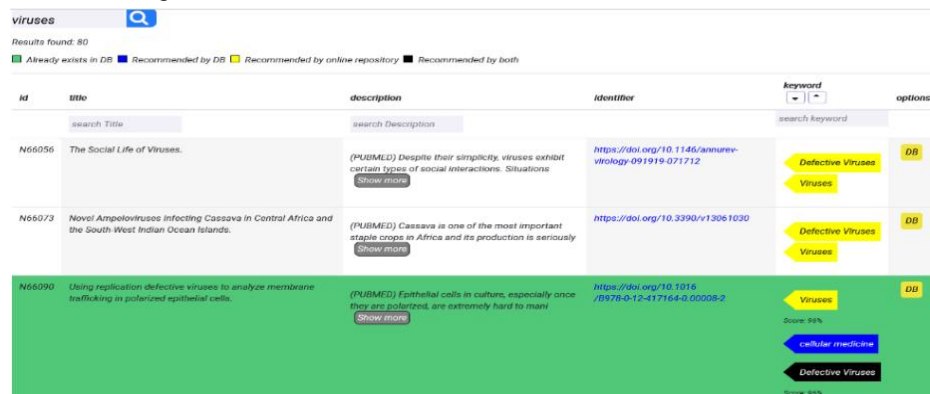
<sup>1</sup> <https://snf-630087.vm.oceanos.grnet.gr:8443/SemanticMiddleware-1.3/results?q={keyword}&validate=true>

**Delete a Learning Object.** Since users have the option to add LOs to the database, they should also be able to remove them. When the user clicks the “delete object” button, a pop-up warning message will appear, notifying him about this irreversible action. When the user confirms the action, the LO and its metadata are removed from the database.

**Update a Learning Object.** Similar to adding, we also provide a variety of features for updating an object. We separated the update feature into 2 sub-features. The first one is responsible for the updating of the least-significant metadata fields of the LO. These include the title, the description as well as the identifier. The second is responsible for updating the most-significant metadata field of the LO, the subject keywords.

## 4.2 Fetching data from external online repository

One of the most important features of the application is fetching LOs from an external source. The user types in the search bar (upper left in Figure 3) the term that wants to see results for. Then, LOs from the external repository are fetched based on their subject keywords. The interface currently accepts external data in XML/OWL or JSON format, provided that they follow the LO Ontology Schema. As documented in [11] however, this is highly configurable and additional mappings can be specified to accommodate any other repository or metadata format. Results are rendered as shown in Figure 3. Selecting the *DB* button under *options* opens a popup window, asking the user to confirm addition to the database. Notice that duplicate objects are highlighted with a different background color.



viruses

Results found: 80

Legend: ■ Already exists in DB ■ Recommended by DB ■ Recommended by online repository ■ Recommended by both

ID	title	description	identifier	keyword	options
N66056	The Social Life of Viruses.	(PUBMED) Despite their simplicity, viruses exhibit certain types of social interactions. Situations <a href="#">Show more</a>	<a href="https://doi.org/10.1146/annurev-virology-091919-071712">https://doi.org/10.1146/annurev-virology-091919-071712</a>	Defective Viruses Viruses	DB
N66073	Novel Ampeloviruses Infecting Cassava in Central Africa and the South West Indian Ocean Islands.	(PUBMED) Cassava is one of the most important staple crops in Africa and its production is seriously <a href="#">Show more</a>	<a href="https://doi.org/10.3390/v13061030">https://doi.org/10.3390/v13061030</a>	Defective Viruses Viruses	DB
N66090	Using replication defective viruses to analyze membrane trafficking in polarized epithelial cells.	(PUBMED) Epithelial cells in culture, especially once they are polarized, are extremely hard to man <a href="#">Show more</a>	<a href="https://doi.org/10.1016/B978-0-12-417164-0.00008-2">https://doi.org/10.1016/B978-0-12-417164-0.00008-2</a>	Viruses cellular medicine Defective Viruses	DB

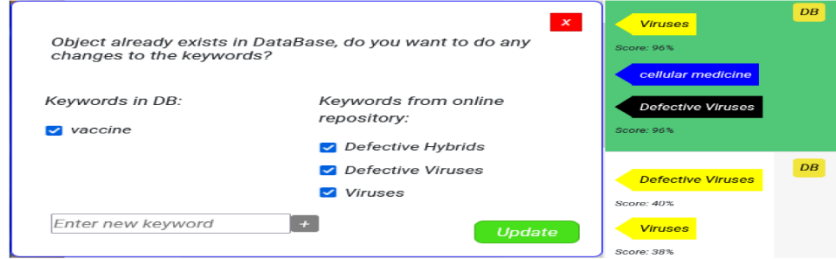
**Fig. 3.** Duplicate LOs and keywords appear with different colors.

A fetched object is marked as duplicate when there is an object in the database with identical title, description, and identifier. When a duplicate LO is selected for addition to the database, an *update current saved object* pop-up window will appear instead (Figure 4). There, the user will be able to choose which keywords to keep and which to reject.

The keywords on the duplicate objects have different colors to stand out. The blue



labeled keywords are the ones already stored in the local DB, the yellow labeled keywords are recommended by the LOOR through semantic matching (Section 3.2) and finally, the black labeled keywords occur in both (Figure 4). For the keywords originating from the LOOR, there is also a validation score available below their lexical representation based on the zero-shot classification procedure described previously.



**Fig. 4.** Left: Update duplicate LO with keywords from external repository. Right: Labels and scores of LOs

## 5 Conclusions and future work

We have presented the design and development of a web interface, through which users are able to interact in a variety of ways with different learning objects and their metadata respectively. Through this process, users can access learning objects in both the interface's database (local repository) and ones in an online, external repository. The ability to produce and validate automated classification suggestions is also integrated into the application. This is combined with collaborative features where authorized users can edit subject annotations and decide on their inclusion to the local pool for others to discover and reuse thus enabling a form of collective intelligence. Finally, our approach can strengthen interoperability, first by providing an infrastructure for harvesting metadata from various sources and then by supporting export of ontology annotations in machine readable format.

As a next step, we intend to conduct a user-centered system evaluation involving target audience (instructors and students) through a series of predefined use-cases and poll user satisfaction. Another improvement would be the addition of a review sub-system. Users will be able to review shown LOs and grade them based on the relevance of the subject they cover and their overall usefulness.

## References

1. Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L. and Stoyanov, V. (2019). Unsupervised cross-lingual representation learning at scale. In Proc. of the 58th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 8440–8451.
2. Daityari, S.: Angular vs React vs Vue: Which Framework to Choose in 2021. Codeinwp (2021). Available: <https://www.codeinwp.com/blog/angular-vs-vue-vs-react/>
3. Downes, S.: New Models of Open and Distributed Learning. In: Jemni M., Kinshuk, Khribi M. (eds) Open Education: from OERs to MOOCs. Lecture Notes in Educational Technology. Springer, Berlin, Heidelberg (2017). DOI: [https://doi.org/10.1007/978-3-662-52925-6\\_1](https://doi.org/10.1007/978-3-662-52925-6_1)
4. d'Oliveira, C., Carson, S. et al.: MIT OpenCourseWare: Unlocking Knowledge, Empowering Minds In: Science, vol. 329, Issue 5991, pp. 525-526 (2010). DOI: 10.1126/science.11826962
5. Europe PMC Consortium.: Europe PMC: a Full-Text literature database for the life sciences and platform for innovation. Nucleic Acids Research. Vol. 43, pp. D1042-D1048 (2015), Database issue PMC.
6. Gasevic, D., Jovanovic, J., Devedzic, V. et al.: Ontologies for reusing learning object content. In: Goodyear, P., Sampson, DG. and Kinshuk DJ et al.(eds.) 5th IEEE international conference on advanced learning technologies (ICALT '05), Kaohsiung, Taiwan, 5–8 July 2005, pp. 944–945. Los Alamitos, CA: IEEE Computer Society Press (2005).
7. Greal, R.: Learning Objects: A practical definition. International Journal of Instructional Technology and Distance Learning. 1(9), 2004. [http://www.itdl.org/Journal/Sep\\_04/article02.htm](http://www.itdl.org/Journal/Sep_04/article02.htm)
8. Guarino, N., Oberle, D., Staab, S.: What is an ontology? In Handbook on ontologies, pp 1–17. Springer (2009).
9. Hodgins, W., Duval, E.: Draft Standard for Learning Object Metadata. Institute of Electrical and Electronics Engineers (2002).
10. Huggingface (2021). Accelerated Inference API (online). Available: <https://api-inference.huggingface.co/docs/python/html/index.html>
11. Koutsomitropoulos, D.: Semantic annotation and harvesting of federated scholarly data using ontologies. Digital Library Perspectives. Vol. 35 No. 3/4, pp. 157-171 (2019). <https://doi.org/10.1108/DLP-12-2018-0038>
12. Koutsomitropoulos, D., Solomou, G.: A learning object ontology repository to support annotation and discovery of educational resources using semantic thesauri. IFLA Journal. 44(1), 2017. DOI: <https://doi.org/10.1177/0340035217737559>
13. Mariano, C.: Benchmarking JavaScript Frameworks. Masters dissertation, 2017. DOI: 10.21427/D72890
14. McMartin, F.: MERLOT: a model for user involvement in digital library design and implementation. Journal of Digital Information. Vol. 5 No. 3 (2006)
15. Roy, D., Sarkar, S., Ghose, S.: A Comparative Study of Learning Object Metadata, Learning Material Repositories, Metadata Annotation & an Automatic Metadata Annotation Tool. Advances in Semantic Computing. Vol. 2, pp. 103-126 (2010)
16. Sabarmathi, G., Chinnaiyan, Dr. R. & Ilango, Dr. V.: Big Data Analytics Research Opportunities and Challenges- A Review. International Journal of Advanced Research in Computer Science and Software Engineering. 6 (10), 227-231(2016)
17. Satrom, B.: Choosing the right JavaScript framework for your next web application. Vitbok RITM0012054. Progress Software Corporation (2018). Available: [https://softarchitect.files.wordpress.com/2018/03/choose-the-right-javascript-framework-for-your-next-web-application\\_whitepaper1.pdf](https://softarchitect.files.wordpress.com/2018/03/choose-the-right-javascript-framework-for-your-next-web-application_whitepaper1.pdf)

18. Ternier, S., Verbert, K., Parra, G., Vandeputte, B., Klerkx, J., Duval, E., Ordonez, V., Ochoa, X.: The ariadne infrastructure for managing and storing metadata. *IEEE Internet Computing*, Vol. 13 No. 4 (2009).
19. U.S. National Library of Medicine. Medical Subject Headings, 2021. Online. Available: <https://www.nlm.nih.gov/mesh/meshhome.html> Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V. and Zettlemoyer, L. (2019). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. arXiv preprint arXiv:1910.13461
20. Yin, W., Hay, J., & Roth, D. (2019). Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. In *Proc. of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP 2019)*, pp. 3914-3923.