JOURNAL OF LEARNING ANALYTICS

SOLAR SOCIETY for LEARNING ANALYTICS RESEARCH

Volume 10(3), 28–44. https://doi.org/10.18608/jla.2023.7897

NLP-Based Management of Large Multiple-Choice Test Item Repositories

Valentina Albano[1], Donatella Firmani[2], Luigi Laura[3], Jerin George Mathew[4], Anna Lucia Paoletti[5], and Irene Torrente[6]

Abstract

Multiple-choice questions (MCQs) are widely used in educational assessments and professional certification exams. Managing large repositories of MCQs, however, poses several challenges due to the high volume of questions and the need to maintain their quality and relevance over time. One of these challenges is the presence of questions that duplicate concepts but are formulated differently. Such questions can indeed elude syntactic controls but provide no added value to the repository.

In this paper, we focus on this specific challenge and propose a workflow for the discovery and management of potential duplicate questions in large MCQ repositories. Overall, the workflow comprises three main steps: MCQ preprocessing, similarity computation, and finally a graph-based exploration and analysis of the obtained similarity values. For the preprocessing phase, we consider three main strategies: (i) removing the list of candidate answers from each question, (ii) augmenting each question with the correct answer, or (iii) augmenting each question with all candidate answers. Then, we use deep learning–based natural language processing (NLP) techniques, based on the Transformers architecture, to compute similarities between MCQs based on semantics. Finally, we propose a new approach to graph exploration based on graph communities to analyze the similarities and relationships between MCQs in the graph. We illustrate the approach with a case study of the *Competenze Digitali* program, a large-scale assessment project by the Italian government.

Notes for Practice

- Established knowledge: Multiple-choice questions (MCQs) are a commonly used format in educational assessments. However, managing large repositories of MCQs can be challenging, especially when trying to avoid duplicate questions. Existing approaches rely on simple lexical-based methods that may not capture the semantic similarity between questions.
- Contribution of the paper: This paper proposes a learning analytics tool that leverages deep learning–based natural language processing (NLP) techniques to compute the semantic similarity between MCQs. The tool allows users to visualize the similarity network of questions and set a threshold to identify possibly duplicate questions. Additionally, the paper proposes three strategies for processing MCQs before computing their similarity: stripping the list of candidate answers, enriching each question with the correct answer, and augmenting each question with all candidate answers.
- Implications for practice: The proposed tool provides educators with a powerful means to manage large repositories of MCQs and mitigate the problem of duplicate questions. This can improve the effectiveness of assessments by ensuring that each question meaningfully contributes to the evaluation of student knowledge. Additionally, the tool can be used to match questions with areas of the syllabus, aiding in curriculum mapping and assessment design. Finally, the paper's use of deep learning–based NLP techniques offers new opportunities to advance the development of educational assessment tools.

Keywords

Multiple-choice question management, natural language processing, deep learning, similarity computation, graph visualization, learning analytics

Submitted: 18/10/2022 — **Accepted:** 11/11/2023 — **Published:** 15/12/2023

[1] *Email: v.albano@governo.it Address: Dip. Funzione Pubblica, Corso Vittorio Emanuele II, 116, 00186 Rome, Italy.*

[2] *Email: donatella.firmani@uniroma1.it Address: Sapienza University, Piazzale Aldo Moro, 5, 00185 Rome, Italy. ORCID ID: https://orcid.org/0000-0003-0358-3208*

[3] *Corresponding author Email: luigi.laura@uninettunouniversity.net Address: Uninettuno University, Corso Vittorio Emanuele II, 39, 00186 Rome,*

Italy. ORCID ID: https://orcid.org/0000-0001-6880-8477
[4] *Email: mathew@diag.uniroma1.it Address: Sapienza University, Piazzale Aldo Moro, 5, 00185 Rome, Italy. ORCID ID: https://orcid.org/0000-0002-4626-826X*
[5] *Email: a.paoletti@funzionepubblica.it Address: Dip. Funzione Pubblica, Corso Vittorio Emanuele II, 116, 00186 Rome, Italy.*
[6] *Email: itorrente@formez.it Address: Formez, Viale Marx, 15, 00137 Rome, Italy.*

1. Introduction

Multiple-choice questions (MCQs) are a popular choice for knowledge assessment in several contexts, ranging from university admission to candidate evaluation for a job position, from self-assessment to game shows like *Who Wants to Be a Millionaire?* and several successful apps for mobile gaming, such as QuizDuello.

Many large-scale standardized tests use MCQs (items) that typically contain four response options, where one option is the correct response (item key) and the other three are the incorrect responses (distractors).

Academic research about MCQs focuses naturally on their effectiveness as a tool for evaluation; from a learning analytics (LA) point of view, we refer to the work of Azevedo and colleagues (2019), where the authors focused on the problem of finding appropriate forms of analysis of MCQs to obtain an assessment method that is as fair as possible for the students. We recall that a commonly accepted definition of LA is “the measurement, collection, analysis and reporting of data about learners and their contexts, for purposes of understanding and optimising learning and the environments in which it occurs” (Long et al., 2011).

Thus, in the context of LA, we also cite a research area focused on the automatic generation of MCQs, since, as observed for example in Mitkov and colleagues (2006), their manual creation is a laborious and time-consuming task. The approaches, based on natural language processing (NLP), focused on several different techniques, such as the use of WordNet, shallow parsing, corpora, ontologies, and, more recently, deep neural networks.

In this paper we propose a novel approach for a related task—the *maintenance* of a large repository of MCQs—and present a prototypical tool used in a real large-scale assessment project by the Italian government. The tool is based on NLP and graph visualization to support the administrator of the learning system in its dynamic use. Our tool can be effective for

- finding questions that are *too similar*;
- automatically checking the coherence between a question and the area of the syllabus it belongs to;
- supporting the *refactoring* of a syllabus, i.e., when the syllabus changes and thus the questions need to be rearranged properly; and
- supporting the merging of two repositories by matching the imported questions to the new syllabus they are moved to.

The results suggest that NLP-based techniques can be beneficial for identifying conceptually duplicated MCQs compared to traditional frequency-based methods, particularly when it comes to questions having the same semantics but different wording. Furthermore, our proposed workflow integrates graph exploration based on graph communities, which presents a novel avenue for comprehending the intricate similarities among MCQs. Our graph exploration step proved helpful in understanding the complex relationships between duplicate questions in our case study. Our proposed approach augments the existing theoretical framework by introducing a practical means of visualizing and analyzing the complex interconnections within large-scale MCQ repositories.

An earlier version of this paper by Albano and colleagues (2022) was presented at IV 2022. Compared to the conference version, this paper has the following features:

- It includes a new method of finding pairs of similar questions in an MCQ repository based on recent deep neural network architectures, which leverages deep learning–based NLP techniques using the Transformers architecture to find similarities based on semantics instead of merely looking at the co-occurrence of the same words in two questions. This approach is expected to find more positives but can also potentially produce false positives, which are pairs of questions that are not truly similar but are identified as such.
- It presents a new graph exploration approach based on graph communities, as opposed to connected components, to handle the possibility of false positives. This method can potentially help to reduce the number of false positives by only considering the similarity between questions within the same community. By doing so, the method can better capture the underlying structure of the MCQ repository and identify clusters of related questions.
- It introduces new application settings for identifying similar records, including options to strip candidate answer lists from each question or to enrich each question with the correct answer only. The first setting may be helpful in scenarios where

the list of candidate answers may contain misleading or irrelevant information, while the latter can provide additional information for understanding the context and meaning of each question.

This paper is organized as follows. The next section provides the necessary background, while Section 3 presents our approach, which is then detailed in Section 4 using a concrete example: the *Competenze Digitali* program, i.e., a large-scale assessment project of the Italian government. In this section we present the results we obtained using our prototypical tool, developed in Python, using Scikit-learn, the Transformers library (Wolf et al., 2020), and the Sentence Transformers library (Reimers & Gurevych, 2019) for the NLP part and Networkx library for graph management.

2. Related Work

The use of MCQs as an assessment tool is garnering increasing interest within the modern educational landscape (Ha & Yaneva, 2018; Kumar et al., 2023). MCQs have found extensive application in various contexts, including college admissions and job placements, serving as a guiding factor in decision-making processes. Despite their unquestionable advantages in knowledge evaluation, MCQs present certain challenges, such as designing questions that align with course objectives and outcomes (Tarrant & Ware, 2012), as well as the potential for answer memorization through the repetition of MCQ stems (i.e., questions), thereby undermining the validity of the examination (Wood, 2009). As mentioned in the introduction, so far the LA field of research has focused, with respect to multiple-choice tests, mainly on their effectiveness as a tool for evaluation; in particular, as observed by Azevedo and colleagues (2019), there are two prominent theories regarding the analysis of questions in assessment tests: the classical test theory (CTT) and the item response theory (IRT). In the CTT the unit of analysis is the overall test, while in the IRT it is the single item.

Several techniques have been used in NLP, including the vector space model (Jain et al., 2019); word2vec (Goldberg & Levy, 2014) and the related word embedding (Bojanowski et al., 2017); semantic features (Salloum et al., 2020); topic modelling (Jelodar et al., 2019); and deep neural networks, used to generate embeddings (Samarinas & Zafeiriou, 2019) and to mimic several of the other approaches (see, e.g., the survey of Kanwal et al., 2021).

More related to our line of research, we mention the approaches aimed to automatically generate MCQs using some of the NLP techniques mentioned above, including ontologies (Papasalouros et al., 2008), machine learning (ML)-based approaches (Ch & Saha, 2018), and hybrid approaches that combine ontologies and ML (Kumar et al., 2023). Additionally, there is the related area of automatic answering of MCQs, which has been investigated through word occurrence–based approaches (Martinez-Gil et al., 2019) as well as more recent ML and deep learning techniques (Chaturvedi et al., 2018; Khashabi et al., 2020).

The problem of identifying duplicate entries (that are, in this case, questions) in a database is generally referred to as entity resolution (Getoor & Machanavajjhala, 2012). In traditional data management applications, the most popular approaches include supervised methods based on learning a vector representation of database records and their text attributes (Brunner & Stockinger, 2020; Ebraheem et al., 2018; Mudgal et al., 2018). In this paper, we focus instead on similarity computation and leave the final decision to a human expert, in the spirit of recent oracle-based approaches such as Firmani and colleagues (2018) and Galhotra and colleagues (2021).

3. Managing Large MCQ Repositories: The Workflow

In this section we detail our proposed approach for the management of large MCQ repositories. The key to the whole approach is the computation of the similarities between the questions; after this computation we (i.e., the experts) need to do some (simple) exploratory data analysis in order to find the threshold value σ of the similarity, i.e., the value that "separates" similar questions from non-similar questions. Deciding the threshold value σ is a choice that will affect all the subsequent steps: a small value of σ will result in several pairs of similar questions, while a high value might result in few pairs of them.

In order to decide on a *reasonable* value of σ, we have two tools: we can plot the distribution of the similarity values in order to get an idea of the cardinality, or we can inspect, for a given pair of questions, the actual text of the questions in order to decide whether or not we believe them to be similar.

Once we select a value of σ, we can build the graph G_σ, i.e., the graph whose nodes are the questions, where there is an edge between two nodes n_i and n_j (corresponding to the questions q_i and q_j) if and only if the similarity of q_i and q_j is greater than σ.

Finally, we can visually explore the graph in order to understand the potential problem between the questions. In this case, it is particularly relevant to focus on the connected components of the graph, i.e., clusters of nodes that are connected and, thus, are similar to each other. In the following section we provide some examples.

Summing up, the workflow proposed is the following:

1. SIMILARITY COMPUTATION: Compute the similarity values between all pairs of questions. This step is performed in an unsupervised manner, meaning that it does not depend on a labelled set of questions with known similarity values to determine the similarity between the questions in each pair.

2. THRESHOLD DEFINITION: Based on the distribution of similarity values, define (i.e., choose) a threshold value σ.

3. GRAPH CONSTRUCTION: Create the graph G_σ, whose nodes correspond to the questions, where there is an edge between two nodes if and only if the similarity of the corresponding questions is greater than σ.

4. GRAPH EXPLORATION: Use graph visual analysis to explore the relationships between the questions.

4. Managing Large MCQ Repositories: An Example

In this section, we present an illustrative application of the workflow described in the previous section. Our aim is to investigate and answer three specific research questions using the proposed methodology.

- R1: Can NLP techniques, in particular language models, be used to compute semantic similarities between MCQs?

- R2: What are the most effective preprocessing strategies for MCQs prior to similarity computation: removing the list of candidate answers, augmenting each question with the correct answer, or augmenting each question with all the answers?

- R3: Can graph exploration techniques, specifically graph communities, assist in identifying potential issues and improving the quality of the repository?

In this section we provide an example using the MCQ database of the *Competenze Digitali* program; this program aims to provide public employees (non-IT specialists) with personalized training, in e-learning mode, on basic digital skills, starting from a structured and homogeneous survey of training needs, in order to increase involvement and motivation, performance, and diffusion and quality of online services, simply and quickly, for citizens and businesses. The implementation of the program is based on the following elements:

- the syllabus that describes the minimum skills required for public employees to operate in an increasingly digital PA ("Pubblica Amministrazione," i.e., Public Administration);

- the online platform that supports processes for detecting individual skills gaps, defining training courses and providing training; and

- the catalogue of quality training, thanks to the collaboration of large players, public and private.

The MCQ dataset of the *Competenze Digitali* program comprises 798 Italian-language questions, each presenting four candidate answers, of which only one is correct. Every question corresponds to a particular syllabus, which groups together questions related to the same topic (e.g., computer networks). In total, there are 11 distinct syllabi in the dataset, with their respective question counts listed in Table 1. We also provide a sample question in Figure 1. Please note that this question and the answers are based on a question that is similar to one found in the MCQ dataset used for this study. However, in order to maintain the privacy and confidentiality of the dataset, the original question and its corresponding answers cannot be provided.

Moreover, our MCQ dataset lacks labels, meaning that we do not have access to a definitive measure of similarity (binary or real valued) for each question pair, indicating their true level of similarity.

Table 1. Number of Questions in the MCQ Database for Each Syllabus

Syllabus	S1.1	S1.2	S1.3	S2.1	S2.2	S3.1	S3.2	S4.1	S4.2	S5.1	S5.2	**Total**
Count	70	94	50	68	134	49	40	57	97	62	77	798

Question: What is the definition of “cookies”?

A. Small files that store information about online browsing on the user’s computer or device.
B. Small files that track and collect user’s online behaviours for targeted advertising purposes.
C. Tokens generated by web applications to authenticate user sessions and enable personalized features.
D. Privacy settings that allow users to control the information shared with websites they visit.

Figure 1. Sample question and answers inspired by a similar question in the MCQ dataset. For clarity, the question has been translated into English.

Similarity Computation. In our case, as mentioned before, we use and compare the two representative methods below:

- the method `CountVectorizer` from the Scikit-learn library (Pedregosa et al., 2011); we used this library because it can be considered as de facto a standard for a large number of machine learning tasks, including basic NLP, and it provides a rich and well-designed API (Buitinck et al., 2013); and
- Multilingual Sentence Transformers models from the Sentence Transformers library (Reimers & Gurevych, 2019); these models can handle multiple languages and can encode input sentences into dense vectors that capture semantic meaning, making them particularly effective for tasks involving sentence similarity and text classification, regardless of the language used.

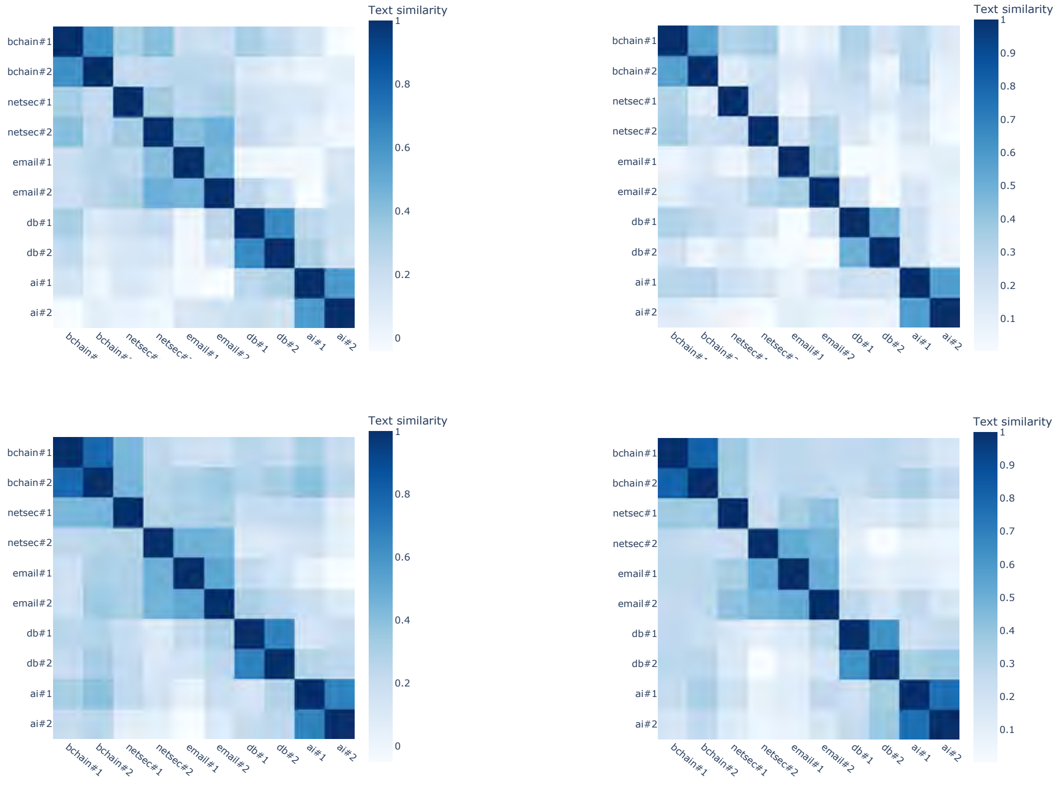
We now describe in more detail the Sentence Transformers models used in the experiments. We considered four multilingual models that were fine-tuned with Italian language data, thus allowing them to handle Italian-language texts with higher accuracy.

- The first one, dubbed `ST-Roberta`, is a variant of the XLM-RoBERTa transformer model (Conneau et al., 2019) that was initially fine-tuned on a large-scale paraphrase dataset consisting of more than 50 languages (Reimers & Gurevych, 2020) and then further fine-tuned for English and Italian. The model is available in the Huggingface repository under the name `T-Systems-onsite/cross-en-it-roberta-sentence-transformer`.
- The second model, dubbed `ST-DistilUSE`, is a multilingual knowledge-distilled version of the multilingual Universal Sentence Encoder (mUSE) (Yang et al., 2019; Reimers & Gurevych, 2020) that supports 15 languages, including Italian. The model is listed as `distiluse-base-multilingual-cased-v1` in the Huggingface repository.
- The third model, named `ST-MiniLM`, is a multilingual variant of MiniLM (Wang et al., 2020) that has been fine-tuned on a vast collection of paraphrase data spanning over 50 languages (Reimers & Gurevych, 2020). The model can be found as `paraphrase-multilingual-MiniLM-L12-v2` in the Huggingface repository.
- Finally, the fourth model, dubbed `ST-MPNet`, is a multilingual variant of MPNet (Song et al., 2020) that has been trained on parallel data for more than 50 languages (Reimers & Gurevych, 2020). The model is available in the Huggingface repository as `paraphrase-multilingual-mpnet-base-v2`.

In our experiments, we considered the four above-listed Sentence Transformers models that were fine-tuned on Italian language data, because this was the specific language of interest. However, in general, the model selection phase can have a significant impact on the quality of the results. For instance, a model trained on legal documents may perform better on legal texts, while a model trained on social media data may perform better on social media texts. Therefore, it is essential to consider the specific language and domain of the text and choose the most appropriate model accordingly.

From this initial selection of Sentence Transformers models we empirically evaluated their performance on a subset of our MCQ dataset and selected the one that achieved the best results. The subset of data used in our experiments was created by handpicking pairs of similar questions (note that these questions are not duplicates) that were related to the same syllabus. We recall that each syllabus focuses on a specific topic, such as blockchains. The goal was to evaluate how well the selected models could capture semantic similarities between questions on the same topic. We report the results in Figure 2, where we show the similarity matrix produced by each model on the select pairs of sentences.

We indicate for each question in Figure 2 its corresponding topic (e.g., `ai#1` and `ai#2` are two sentences related to AI). Ideally, we would expect to see a pattern of dark blue squares of size two on the diagonal of the similarity matrix, indicating that each pair of questions related to the same topic has a high similarity score and is also dissimilar from the other questions in the dataset. However, the results on the subset of data varied across different models. For instance, `ST-Roberta` and `ST-DistilUSE` gave a modest similarity score (ranging from 0.55 to 0.61) to the first pair of questions related to blockchains (i.e., `bchain#1` and `bchain#2`) compared to `ST-MiniLM` and `ST-MPNet`, which were more prone to give higher scores



(c) *ST-MiniLM*

(d) *ST-MPNet*

Figure 2. Similarity matrices produced by each Sentence Transformers model on the handpicked dataset of similar question pairs.

to questions related to the same topic. After comparing the results across different models, we observed that `ST-MPNet` performed better than the others, and hence we chose it for the subsequent stages of our pipeline.

It is important to mention that, irrespective of the choice between the above similarity methods, there are different approaches for computing the similarity of the questions in the case of multiple-choice test items, based on the following observation: whether the text of the question should include the possible answers or not. We believe this choice depends on the repository (and, in some sense, on the domain); for example, our repository included several questions whose text was simply "Which of the following statement is false?," which, without adding the four response options, are indeed completely identical.

We experimented with three approaches, dubbed question-only, all-answers, and correct-only.

1. **Question-only.** The text for similarity computation is only the text of the question.
2. **All-answers.** The text for similarity computation is the text of the question together with the texts of the four answers provided.
3. **Correct-only.** The text for similarity computation is the text of the question together with the text of the correct answer.

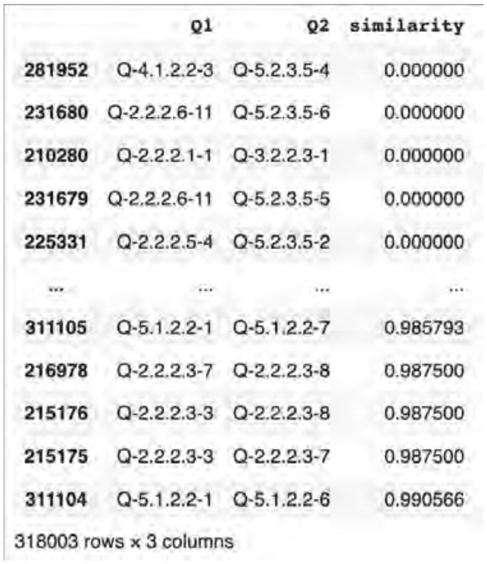
In the following, when not stated explicitly, we report the results of the second case, i.e., the all-answers approach.

In our case, we had 798 MCQs in our repository; thus the number of pairs of questions is $798 \cdot (798-1)/2 = 318{,}003$.

In Figure 3 we can see the similarity values sorted by their value; as we expected, at the top of the table we have completely unrelated questions (similarity equal to zero) and, at the bottom, we can spot some suspicious items that are definitely too similar (similarity values in the range 0.98–0.99). Since, as we mentioned before, we have more than three hundred thousand similarity values, we need to analyze them using some plots, as we see in the following.

Threshold definition. In Figures 4 to 6 we show the distribution of similarity values for all the pairs of questions computed by the five similarity computation methods (`CountVectorizer`, `ST-Roberta`, `ST-DistilUSE`, `ST-MiniLM`, and `ST-MPNet`) and the three approaches considered (question-only, all-answers, correct-only).

In particular, Figure 4 reports the distribution for the `CountVectorizer` method in the all-answers approach. We can see that approximately two hundred thousand pairs have a similarity score less than 0.2.



	Q1	Q2	similarity
281952	Q-4.1.2.2-3	Q-5.2.3.5-4	0.000000
231680	Q-2.2.2.6-11	Q-5.2.3.5-6	0.000000
210280	Q-2.2.2.1-1	Q-3.2.2.3-1	0.000000
231679	Q-2.2.2.6-11	Q-5.2.3.5-5	0.000000
225331	Q-2.2.2.5-4	Q-5.2.3.5-2	0.000000
...	...	...	...
311105	Q-5.1.2.2-1	Q-5.1.2.2-7	0.985793
216978	Q-2.2.2.3-7	Q-2.2.2.3-8	0.987500
215176	Q-2.2.2.3-3	Q-2.2.2.3-8	0.987500
215175	Q-2.2.2.3-3	Q-2.2.2.3-7	0.987500
311104	Q-5.1.2.2-1	Q-5.1.2.2-6	0.990566

318003 rows × 3 columns

Figure 3. Question similarities sorted by values (CountVectorizer, all-answers).

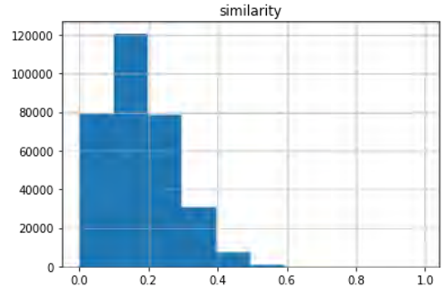
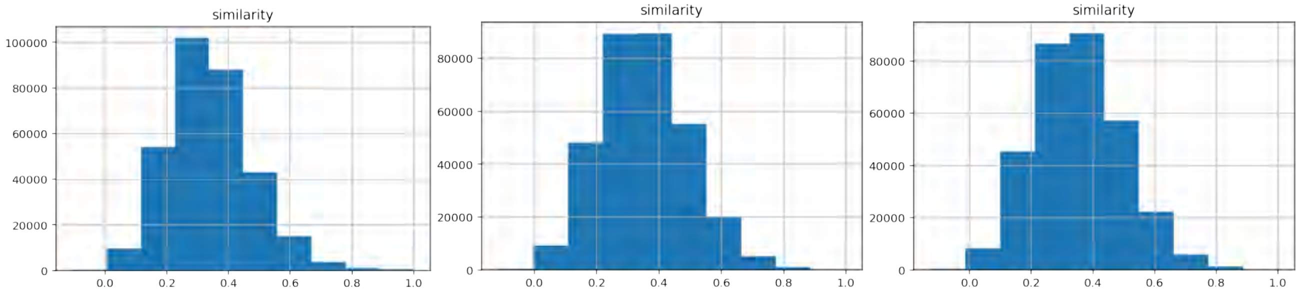


Figure 4. Similarity value distribution (CountVectorizer, all-answers).



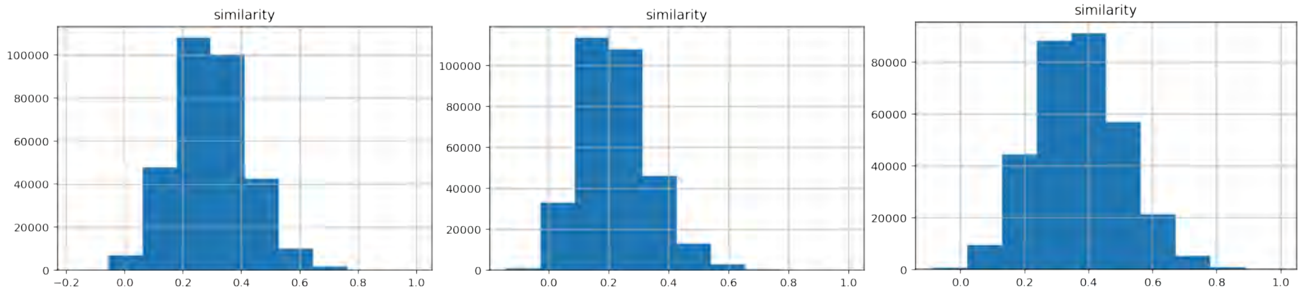
(a) *question-only* **(b)** *all-answers* **(c)** *correct-only*

Figure 5. Similarity value distribution (ST-MPNet).

Figure 5 shows the similarity value distribution for the ST-MPNet methods in the three approaches considered (question-only, all-answers, correct-only). With respect to CountVectorizer we have generally higher similarity scores, because ST-MPNet can better capture synonyms (e.g., database management system and DBMS) and cross-language references (e.g., Open Data and Dati Aperti).

Finally, in Figure 6, we plot the results for the other selected models from the Sentence Transformers library (all-answers). The similarity distribution of these methods is analogous to ST-MPNet.

From now on we consider CountVectorizer because it is simple and ST-MPNet because it performed better compared to the other models. In Table 2 we have a more detailed picture of the numbers for these methods. In particular, for the CountVectorizer method we can see that a large majority ($\approx 97\%$) of pairs have a similarity score less than 0.4, and almost all the pairs ($\approx 99\%$) have a similarity score less than 0.5. For ST-MPNet, the same buckets contain $\approx 70\%$ and $\approx 88\%$



(a) `ST-Roberta` **(b)** `ST-DistilUSE` **(c)** `ST-MiniLM`

Figure 6. Similarity value distribution (all-answers).

Table 2. Similarity Value Distribution for Different Values of σ

Value	# of pairs	percentage
$0 <= \sigma < 0.4$	309,379	97.29%
$0.4 <= \sigma < 0.5$	7,192	2.26%
$0.5 <= \sigma < 0.6$	1,084	0.34%
$0.6 <= \sigma < 0.7$	125	0.04%
$0.7 <= \sigma < 1.0$	223	0.07%
total	318,003	

(a) `CountVectorizer`, all-answers

Value	# of pairs	percentage
$0 <= \sigma < 0.4$	222,310	69.91
$0.4 <= \sigma < 0.5$	57,791	18.17
$0.5 <= \sigma < 0.6$	25,335	7.97
$0.6 <= \sigma < 0.7$	8,786	2.76
$0.7 <= \sigma < 1.0$	3,781	1.19
total	318,003	

(b) `ST-MPNet`, question-only

Value	# of pairs	percentage
$0 <= \sigma < 0.4$	205,771	64.71%
$0.4 <= \sigma < 0.5$	64,958	20.43%
$0.5 <= \sigma < 0.6$	32,564	10.24%
$0.6 <= \sigma < 0.7$	10,874	3.42%
$0.7 <= \sigma < 1.0$	3,836	1.21%
total	318,003	

(c) `ST-MPNet`, all-answers

Value	# of pairs	percentage
$0 <= \sigma < 0.4$	204,254	64.23%
$0.4 <= \sigma < 0.5$	64,310	20.22%
$0.5 <= \sigma < 0.6$	33,182	10.43%
$0.6 <= \sigma < 0.7$	11,897	3.74%
$0.7 <= \sigma < 1.0$	4,360	1.37%
total	318,003	

(d) `ST-MPNet`, only-correct

of the pairs, respectively, with more than $\approx 10\%$ of pairs having similarity larger than or equal to 0.5.

For the subsequent steps, we consider as thresholds the natural value 0.5 and a slightly higher value 0.7. In Figure 7, we show the filtered similarity value distributions for the `CountVectorizer` and `ST-MPNet` methods with respect to the two threshold values.

Graph Construction with `CountVectorizer`. As we mentioned before, we refer to two different values of the threshold σ, obtaining two different graphs. Indeed, we should aim at one threshold value and one graph, but here we want to provide two different examples. In particular, we now consider the similarity values computed by `CountVectorizer` and build

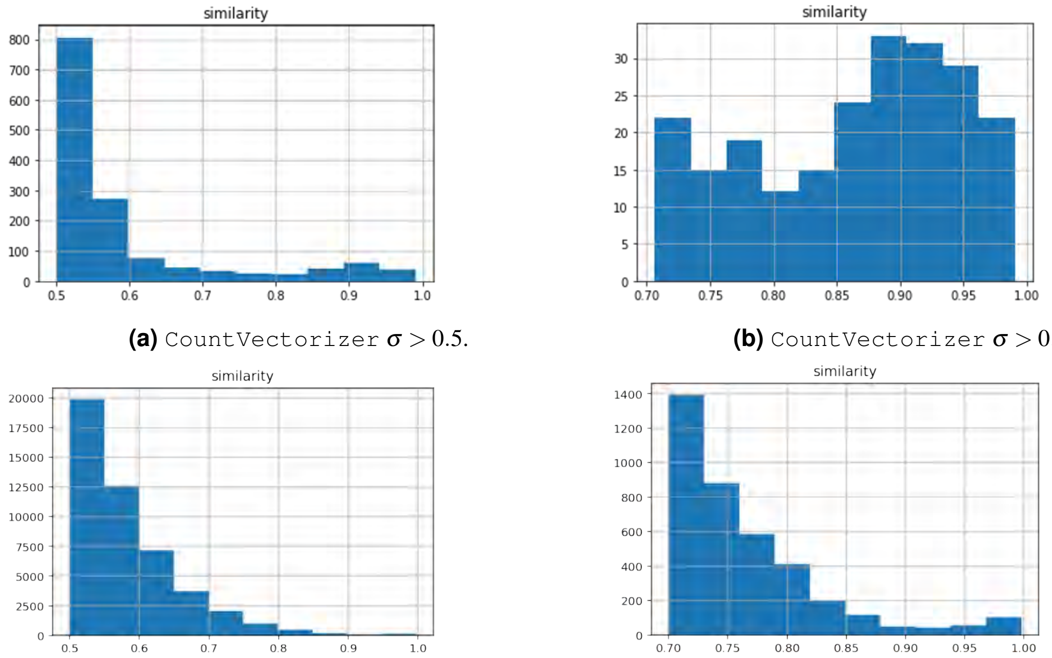
- G_5, with all the edges whose similarities values are > 0.5, and
- G_7, with all the edges whose similarities values are > 0.7.

Note that, by definition, all the edges of G_7 are also edges of G_5.

Note that we do not consider all the questions as nodes of the graph, but only the ones that have at least one edge, i.e., the ones that have a similarity value (greater than the threshold) with another question. Thus, our graphs do not have 798 nodes, but G_5 has 569 nodes and 1430 edges, and G_7 has 296 nodes and 223 edges.

We can now visually explore the similarities of the questions in the repository. Let us start with G_5, shown in Figure 8a. As we can see, this graph is still (in this context) moderately large, with more than a thousand edges. If we focus on its structure, we find that it has 66 connected components (CC); the largest CC is shown in Figure 8b: it has 384 nodes and 1271 edges; indeed, looking at Figure 8a, we can see that the graph is made by a single large connected component and several tiny ones.

Now let us focus on G_7, shown in Figure 9a. In this case we have a relatively small graph, made of several tiny connected components; indeed, it has 122 CCs, and the largest one, shown in Figure 9b, is small: it has only seven nodes and nine edges.



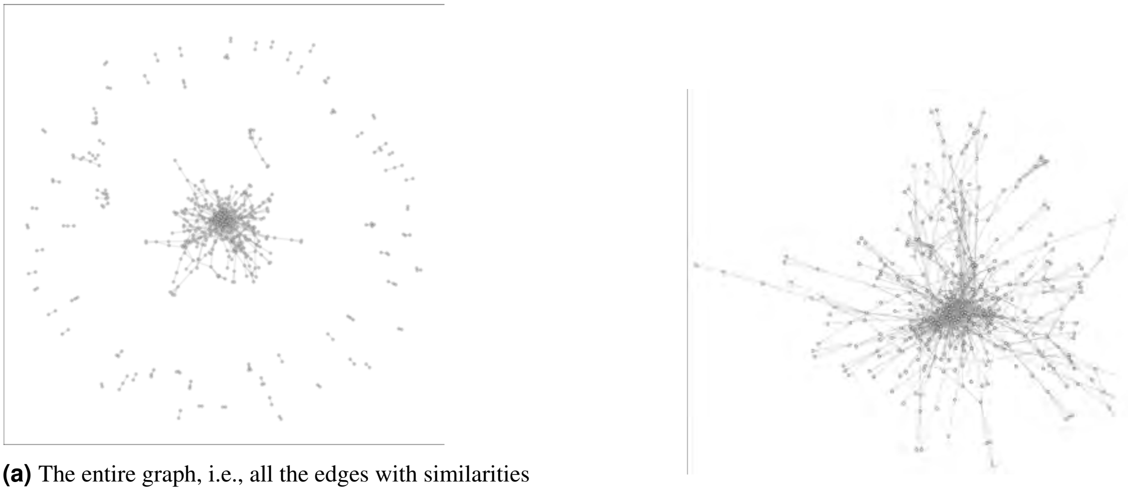
(a) `CountVectorizer` $\sigma > 0.5$.

(b) `CountVectorizer` $\sigma > 0.7$.

(c) `ST-MPNet` $\sigma > 0.5$.

(d) `ST-MPNet` $\sigma > 0.7$.

Figure 7. Filtered similarity value distribution (all-answers).

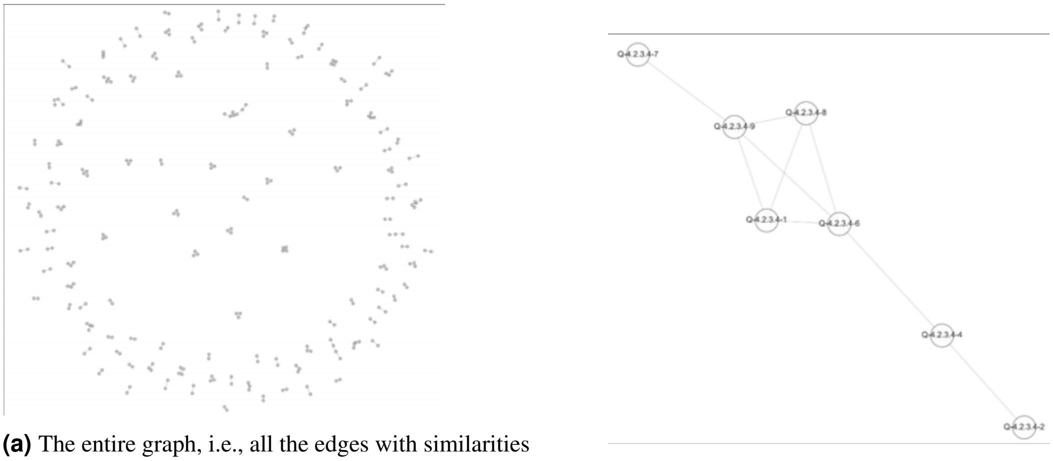


(a) The entire graph, i.e., all the edges with similarities $\sigma > 0.5$.

(b) The largest connected component.

Figure 8. Topology layout of G_5 (`CountVectorizer`, all-questions.
)

Graph Construction with `ST-MPNet`. We now follow the same graph construction protocol but use the similarity values computed by the `ST-MPNet` method. The resulting G_5 graph has 797 nodes and $47,274$ edges, while G_7 has 718 nodes and 3836 edges. Given the higher similarity values computed by `ST-MPNet` than by `CountVectorizer`, both G_5 and G_7 are much more connected, with one and 30 connected components, respectively. Also, the largest CCs are larger, with 797 and 614 nodes, respectively. In this scenario, the CC-based analysis may be too coarse, and further decomposition may be needed. To address this, we show in Figure 10 the result of the traditional Clauset–Newman–Moore community detection algorithm (Clauset et al., 2004) on the largest CC of G_7. In many applications, the communities resulting from the latter step are fine grained enough to be visually explored, and groups of similar questions can be identified manually. Otherwise, it is also possible to apply node-centrality methods as in Ausiello and colleagues (2013, 2012) to identify salient questions and boost visual exploration.



(a) The entire graph, i.e., all the edges with similarities $\sigma > 0.7$.

(b) The largest connected component.

Figure 9. Topology layout of G_7 (`CountVectorizer`, all-questions).

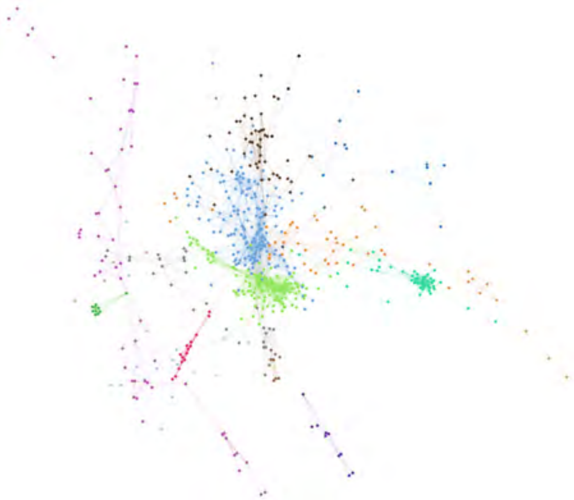


Figure 10. The largest CC in G_7 (`ST-MPNet`, all-answers) with node colours highlighting the community structure.

5. Application Tasks

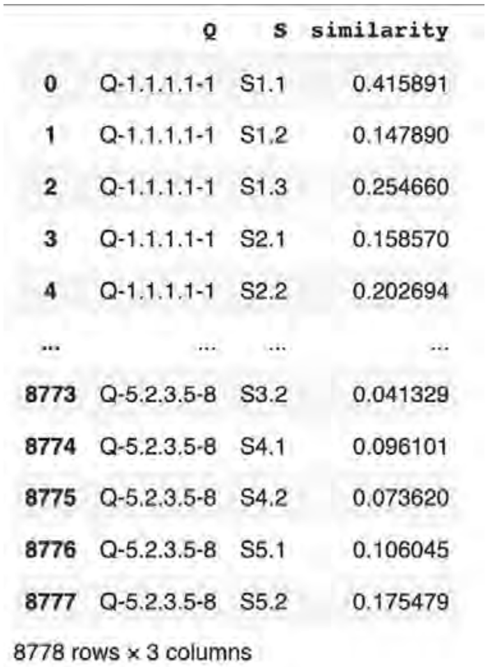
Now we discuss some applications of the above-mentioned approach.

5.1 Syllabus Coherence

Using the similarity techniques described before, it is also possible, if a syllabus is available, to check the coherence of the questions to specific areas of the syllabus. In the case of the *Competenze Digitali* program, our syllabus is made of five (major) areas and 11 sub-areas; in particular, the first area has three sub-areas (denoted by S1.1, S1.2, and S1.3), while the other four areas have only two sub-areas (e.g., for the second area we have S2.1 and S2.2). The questions in the database have an unique identifier that starts with the number of the area and the number of the sub-area.

Using this, we can easily check whether a question is similar to the (text description of the) sub-area of the syllabus it belongs to. In our dataset we have 798 questions and 11 sub-areas, so we computed $798 \cdot 11 = 8778$ values of similarity. Look at Figure 11, where we can see the head and the tail of the values; the first 11 items (we can see only five of them) are the similarity score of the first question in the dataset against the first five sub-areas of the syllabus; it is easy to see that the largest score among the ones shown in the table is exactly for the sub-area of the syllabus it belongs to (i.e., Q-**1.1**.1.1-1 belongs to S**1.1**), and the same happens for the last question (i.e., Q-**5.2**.3.5-8 belongs to S**5.2**).

In our dataset, approximately half of the questions have the largest similarity score, among the 11 sub-areas of the syllabus, with the one they belong to.



	Q	S	similarity
0	Q-1.1.1.1-1	S1.1	0.415891
1	Q-1.1.1.1-1	S1.2	0.147890
2	Q-1.1.1.1-1	S1.3	0.254660
3	Q-1.1.1.1-1	S2.1	0.158570
4	Q-1.1.1.1-1	S2.2	0.202694
...	...	...	...
8773	Q-5.2.3.5-8	S3.2	0.041329
8774	Q-5.2.3.5-8	S4.1	0.096101
8775	Q-5.2.3.5-8	S4.2	0.073620
8776	Q-5.2.3.5-8	S5.1	0.106045
8777	Q-5.2.3.5-8	S5.2	0.175479

8778 rows × 3 columns

Figure 11. The similarity values between the questions and the areas of the syllabus.

This finding shows the ability of the proposed approach to evaluate the coherence between questions and the syllabus, enabling educators to ensure alignment between assessment items and specific areas of the curriculum. By examining the similarity scores and the corresponding sub-areas of the syllabus, educators can identify areas that may require additional focus or clarification (e.g., lack of questions related to a specific syllabus). Moreover, the evaluation of coherence between questions and the syllabus also helps educators identify potential redundancies in their question sets. By comparing the similarity scores across different sub-areas, educators can pinpoint instances where multiple questions assess the same or similar concepts.

In these scenarios, leveraging modern NLP-based models (such as deep learning architectures) instead of relying solely on lexical approaches can effectively allow such use cases. By considering the semantic and contextual relationships between words and phrases, NLP models can provide a more comprehensive understanding of the content and meaning of the text than our previous approach, which was limited to the co-occurrence of words in both questions and the syllabus.

5.2 Syllabus Refactoring

Our proposed approach is also useful in the syllabus refactoring process, which involves making changes to one or more parts of the syllabus. In this context, the challenge lies in identifying the questions that pertain to the modified parts of the syllabus. In contrast to our previous work on similarity methods based on word occurrence, our new approach incorporates modern NLP models, allowing us to analyze the textual content of both the modified syllabus components and the questions in the repository to accurately measure semantic similarity and capture more nuanced relationships between the two. With the assistance of our approach, educators can easily identify the questions that are related to the specific parts of the syllabus that have undergone changes. By analyzing the semantic similarities and contextual relevance, the models can accurately pinpoint the questions that align with the modified syllabus sections.

One notable advantage of our approach in syllabus refactoring is its efficiency. Manual methods of tracking and updating questions in response to syllabus changes can be time-consuming and error prone. Our approach can help in identifying and updating the questions that align with the modified syllabus, reducing the burden on educators and ensuring a more streamlined and accurate identification of relevant questions.

5.3 Merging Repositories

The similarity-based approach could also be useful when one has to merge two different MCQ repositories that are based on the same topic but refer to different syllabi: in this case, while a “natural” approach is usually a manual work in which one tries to match the areas of the syllabi, in practice there is usually an overlap in several areas, making it impossible to find a direct correspondence. In this case, it might be more practical to simply compute the similarity between the new set of questions and the areas of the syllabus they are merged into; this approach does not aim to replace human work but to simplify it by a quick preprocessing. NLP-based models can prove useful in this application scenario. Instead of relying solely on lexical similarities or manual matching (as in our previous work), NLP models can consider the underlying meaning and conceptual overlap between questions and syllabus areas. This enables a more accurate assessment of the alignment and facilitates the identification of related content, even when there is no direct correspondence between the two repositories.

6. Discussion

In this section we revisit our three research questions and envision further development, starting from current limitations of the proposed approach and how end users (e.g., educators) could use the proposed approach in a real-world scenario.

6.1 Answers to Research Questions

We now provide answers to our research questions based on our experimental observations.

R1 *Can NLP techniques, in particular language models, be used to compute semantic similarities between MCQs?* Novel language models proved to be a very effective tool to compute semantic similarities between MCQs, thanks to their strong ability to understand the contextual meaning of text, including questions. In contrast, traditional methods such as `CountVectorizer` fail to identify many of the similar question pairs, thus being less valuable in tasks like identifying duplicate questions and question clustering, or evaluating the overall diversity of question sets.

R2 *What are the most effective preprocessing strategies for MCQs prior to similarity computation: removing the list of candidate answers or augmenting each question with the correct answer?* When it comes to preprocessing MCQs for similarity computation, both removing the list of candidate answers and augmenting each question, either with the correct answer or all the answers, have their advantages and considerations. Removing the list of candidate answers can help focus on the core content of the question, reducing potential noise introduced by the options. On the other hand, augmenting each question with the correct answer or all the answers can enhance the semantic representation of the question, providing additional context.

R3 *Can graph exploration techniques, specifically graph communities, assist in identifying potential issues and improving the quality of the repository?* Graph exploration techniques, particularly those involving graph communities, can play a valuable role in identifying potential issues and enhancing the quality of a repository. By representing the repository as a graph, where questions and their similarity are nodes and edges, respectively, communities within the graph can reveal clusters of related or similar questions, aiding in tasks such as identifying redundant or overlapping content.

6.2 Comparison with Similar Works

In this section, we discuss other works that share the objective of identifying duplicate questions and draw comparison with them. The task of finding duplicate questions has been particularly investigated in the Q&A field with the aim of answering queries by finding other similar questions in Q&A forums that have already been answered. One such work is Li and colleagues (2018), which presents a deep learning–based system designed to recognize similar questions in the context of unsolved medical queries. The primary focus of this work is to assist users of medical Q&A platforms in finding answers to health-related questions by identifying duplicate queries that have already been answered. The approach involves training a long short-term memory (LSTM) (Hochreiter & Schmidhuber, 1997) neural network on pairs of questions, mapping semantically analogous queries closer together within the LSTM network's vector space. The system then employs this trained network to embed unanswered queries into vectors, subsequently comparing these vectors with those generated by the LSTM network for previously answered questions on the Q&A platform. Kamienski and colleagues (2023) also propose an analogous concept to Li and colleagues (2018) for Q&A platforms focused on game development. The authors propose the use of a combination of large pretrained deep learning models and unsupervised techniques to identify duplicate questions. Similarly to Li and colleagues (2018), the primary goal is to help game developers find answers to their queries by identifying similar previously posted questions that have been answered. More specifically, the proposed methodology involves using outputs from various unsupervised and semi-supervised methods, including large language models such as MPNet (Song et al., 2020), as features to train a supervised model to predict the similarity score between sentence pairs.

Both Li and colleagues (2018) and Kamienski and colleagues (2023) differ from our work in that they use a supervised approach to learn a model that predicts whether a pair of questions is duplicated or not. In contrast, since we do not have access to the ground-truth labels that denote which MCQs are duplicated, we opted for an unsupervised approach by leveraging existing pretrained large language models to discover duplicate MCQs.

On the other hand, to the best of our knowledge, only a few works have investigated the task of finding duplicate questions in the LA field. In this context, we mention Mukherjee and Kumar (2019), who present a machine learning–based system for managing extensive question paper databases. This system identifies pairs of English sentences exhibiting high semantic similarity by training an XGBoost model (Chen & Guestrin, 2016) with diverse manually chosen features, including structural attributes (e.g., token count) and word embeddings. The training data consist of labelled duplicate question pairs from Quora.

Although sharing similar goals, this work does not incorporate further processing of the model's output to enable accessible navigation of the similarity model's results. In contrast, our approach integrates a graph construction phase intended to simplify duplicate identification and reasoning over model outcomes.

Finally, we mention Mia and Latiful Hoque (2019), who introduce an efficient system for detecting and managing duplicate questions within a Bengali-language MCQ database of a problem-based e-learning (PBeL) system. In contrast to the previously mentioned works, this study does not employ deep learning techniques. Instead, the authors represent each MCQ with a vector via the TF-IDF algorithm, identifying similar MCQs using cosine similarity. We observed in our work that TF-IDF has not proved effective in finding duplicated MCQs.

6.3 Current Limitations

We now discuss limitations of this work and future research directions. Firstly, we acknowledge that selecting the right NLP model may not be straightforward since it depends on various factors such as the language of the dataset and its domain. For instance, the Sentence Transformers models mentioned earlier may not perform optimally for low-resource languages, specific domains like the legal domain, or datasets that have specific characteristics that differ from the *Competenze Digitali* MCQ dataset we used in our example. In such cases, developing custom Sentence Transformers models tailored to the specific language or domain may be necessary to effectively identify pairs of similar questions. Additionally, when multiple models are suitable for the task, defining criteria to select the best-performing model becomes crucial. This may involve manually selecting questions that are known to be duplicates or very similar to assess and compare the performance of different models. Exploring the appropriate model selection process and criteria among different models is an avenue for future research.

Another current limitation of our work is the lack of integration of human annotators within the pipeline. Currently, our approach focuses on identifying pairs of sentences that are likely duplicates or highly similar, with the final decision left to the human expert. However, a more end-to-end strategy could involve integrating human feedback from the initial stages, leveraging both human judgment and the NLP approach to validate and reinforce each other. This approach would involve actively incorporating human annotators into the process, using their expertise to assess and validate the results generated by the NLP models. Investigating the integration of human feedback and collaboration within the pipeline is a promising area for future research.

6.4 Envisioned Workflow for MCQ Management

To fully leverage the proposed approach, we envision the development of a user-friendly interface that will serve as a central platform for managing MCQ repositories. Such an interface will offer various features and functionalities to support educators in their task of finding duplicate questions. One of the key capabilities of the interface will be the ability to load an MCQ dataset, allowing users to easily import their existing question banks. Once the dataset is loaded, educators can browse through the questions, gaining a comprehensive understanding of the available content. To assist users in assessing the similarity between questions, the user interface (UI) will offer a range of similarity measures, such as those introduced in Section 4. Educators can select different measures, such as lexical or semantic similarity, and customize the corresponding thresholds based on their preferences and requirements. Upon selecting the desired similarity measures and thresholds, educators can apply them to a specific set of questions that they have chosen through the UI. This enables them to evaluate the effectiveness of different methods and thresholds on a subset of questions, gaining insight into the best approach for their needs. Finally, the envisioned UI will allow them to find relationships between questions by generating and exploring the graphs of communities within the MCQ repository using our proposed approach. Educators can select different community detection algorithms and interactively explore the graph, examining the connections between questions. This interactive exploration will allow users to manually review potential duplicates and determine their validity.

It is essential to highlight that our approach places the final decision-making authority in the hands of the human user. While the system provides recommendations based on the applied models, thresholds, and community detection algorithms, educators have the ultimate discretion to choose the most suitable model, threshold, and community detection algorithm for their specific context.

In this work, our primary focus was on developing an effective pipeline for identifying duplicate questions within MCQ repositories. Although we recognize the importance of an appropriate UI for facilitating this task, we acknowledge that its design was beyond the scope of this work. Therefore, the development of a user-friendly interface for managing MCQ repositories is left as an avenue for future research.

6.5 Ethical Issues

Our study does not involve human subjects or sensitive data, neither from the participants in the *Competenze Digitali* program nor from the experts compiling the question database. The main ethical aspect that we considered is therefore the confidentiality of the question database itself. However, we can share the similarity scores used in our experiments and, upon request, we can also share the main codebase that implements the described workflow and the demonstrative questions that were used for the examples throughout this paper.

When implementing our workflow in authentic classrooms, the same confidentiality aspect could exist. Since our workflow is meant to be used mainly by educators, it is reasonable to assume that human users of our system would have permission to access the question databases to review the provided recommendations.

Future versions of our workflow could involve data from the experts compiling the question database to mitigate the risk of discriminating against certain user groups when performing the duplicate detection task. Identifying and mitigating potential biases toward protected categories when performing analogous data-cleaning tasks is nowadays an active area in data management, and further analysis in this direction is left as an interesting future work. We refer the interested reader to recent works in the related entity-resolution area (Efthymiou et al., 2021).

7. Conclusions

In this paper, based on our experience in a real large-scale assessment project by the Italian government, we proposed a novel approach for the maintenance of a large repository of MCQs. This paper builds upon a previous work presented at IV 2022 (Albano et al., 2022) and introduces three major additions.

Firstly, a new method for identifying pairs of similar questions in a MCQ repository has been proposed. The approach uses deep learning–based NLP techniques based on recent Transformers architectures to find similarities based on the semantics of the questions, instead of relying on the co-occurrence of words. This method is expected to identify more similar pairs but can also produce false positives.

Secondly, a new graph exploration approach based on graph communities has been proposed to handle the issue of false positives. This method can potentially reduce the number of false positives by considering the similarity only between questions within the same community. By doing so, the method can better capture the underlying structure of the MCQ repository and identify clusters of related questions.

Lastly, new application settings have been introduced for identifying similar records. These settings include options to (1) strip candidate answer lists from each question or to enrich each question either (2) with the correct answer only or (3) all the answers. The first option can be useful in scenarios where the list of candidate answers may contain misleading or irrelevant information, while the other two can provide additional context and meaning for each question.

Our prototypical tool, based on NLP and graph visualization, has been effective for different tasks, including finding questions that are too similar and checking the coherence between a question and the area of the syllabus it belongs to.

Furthermore, this tool could be employed in other tasks, such as the refactoring of a syllabus, i.e., when the syllabus changes and thus the questions need to be rearranged, and the merging of two repositories, by matching the imported questions against the new syllabus they are moved to.

Declaration of Conflicting Interest

The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The work by Donatella Firmani and Jerin George Mathew was partly supported by Sapienza Research Project “Trustworthy Technologies for Augmenting Knowledge Graphs” (grant number B83C22007180001). This work was carried out while Jerin George Mathew was enrolled in the Italian National Doctorate on Artificial Intelligence run by Sapienza University of Rome.

References

Albano, V., Firmani, D., Laura, L., Paoletti, A. L., & Torrente, I. (2022). Managing large multiple-choice test item repositories. In *Proceedings of the 26th International Conference on Information Visualisation* (IV 2022), 19–22 July 2022, Vienna, Austria (pp. 275–279). IEEE. https://doi.org/10.1109/IV56949.2022.00054

Ausiello, G., Firmani, D., & Laura, L. (2012). Real-time monitoring of undirected networks: Articulation points, bridges, and connected and biconnected components. *Networks*, *59*(3), 275–288. https://doi.org/10.1002/net.21450

Ausiello, G., Firmani, D., & Laura, L. (2013). The (betweenness) centrality of critical nodes and network cores. In *Proceedings of the Ninth International Wireless Communications and Mobile Computing Conference* (IWCMC 2013), 1–5 July 2013, Cagliari, Sardinia, Italy (pp. 90–95). IEEE. https://doi.org/10.1109/IWCMC.2013.6583540

Azevedo, J. M., Oliveira, E. P., & Damas Beites, P. (2019). Using learning analytics to evaluate the quality of multiple-choice questions: A perspective with classical test theory and item response theory. *The International Journal of Information and Learning Technology*, *36*(4), 322–341. https://doi.org/10.1108/IJILT-02-2019-0023

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, *5*, 135–146. https://doi.org/10.1162/tacl_a_00051

Brunner, U., & Stockinger, K. (2020). Entity matching with transformer architectures—A step forward in data integration. In *Proceedings of the 23rd International Conference on Extending Database Technology* (EDBT 2020), 30 March–2 April 2020, Copenhagen, Denmark (pp. 463–473). https://doi.org/10.5441/002/edbt.2020.58

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., VanderPlas, J., Joly, A., Holt, B., & Varoquaux, G. (2013). API design for machine learning software: Experiences from the Scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 23 September 2013, Prague, Czechia (pp. 108–122). https://www.ecmlpkdd2013.org/wp-content/uploads/2013/09/lml2013_api_sklearn.pdf

Ch, D. R., & Saha, S. K. (2018). Automatic multiple choice question generation from text: A survey. *IEEE Transactions on Learning Technologies*, *13*(1), 14–25. https://doi.org/10.1109/TLT.2018.2889100

Chaturvedi, A., Pandit, O., & Garain, U. (2018). CNN for text-based multiple choice question answering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics* (ACL 2018), 15–20 July 2018, Melbourne, Australia (pp. 272–277). Association for Computational Linguistics. https://doi.org/10.18653/v1/P18-2044

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (KDD 2016), 13–17 August 2016, San Francisco, California, USA (pp. 785–794). ACM. https://doi.org/10.1145/2939672.2939785

Clauset, A., Newman, M. E. J., & Moore, C. (2004). Finding community structure in very large networks. *Physical Review E*, *70*(6). https://doi.org/10.1103/physreve.70.066111

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., & Stoyanov, V. (2019). Unsupervised cross-lingual representation learning at scale. *arXiv preprint: 1911.02116*. https://doi.org/10.48550/arXiv.1911.02116

Ebraheem, M., Thirumuruganathan, S., Joty, S. R., Ouzzani, M., & Tang, N. (2018). Distributed representations of tuples for entity resolution. *Proceedings of the VLDB Endowment*, *11*(11), 1454–1467. https://doi.org/10.14778/3236187.3236198

Efthymiou, V., Stefanidis, K., Pitoura, E., & Christophides, V. (2021). FairER: Entity resolution with fairness constraints. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management* (CIKM 2021), 1–5 November 2021, Queensland, Australia (online) (pp. 3004–3008). ACM. https://doi.org/10.1145/3459637.3482105

Firmani, D., Galhotra, S., Saha, B., & Srivastava, D. (2018). Robust entity resolution using a CrowdOracle. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, *41*(2), 91–103. http://sites.computer.org/debull/A18june/p91.pdf

Galhotra, S., Firmani, D., Saha, B., & Srivastava, D. (2021). Efficient and effective ER with progressive blocking. *The VLDB Journal*, *30*(4), 537–557. https://doi.org/10.1007/s00778-021-00656-7

Getoor, L., & Machanavajjhala, A. (2012). Entity resolution: Theory, practice & open challenges. *Proceedings of the VLDB Endowment*, *5*(12), 2018–2019. https://doi.org/10.14778/2367502.2367564

Goldberg, Y., & Levy, O. (2014). Word2vec explained: Deriving Mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint: 1402.3722*. https://doi.org/10.48550/arXiv.1402.3722

Ha, L., & Yaneva, V. (2018). Automatic distractor suggestion for multiple-choice tests using concept embeddings and information retrieval. In J. Tetreault, J. Burstein, E. Kochmar, C. Leacock, & H. Yannakoudakis (Eds.), *Proceedings of the 13th Workshop on Innovative Use of NLP for Building Educational Applications*, 5 June 2018, New Orleans, Louisiana, USA (pp. 389–398). Association for Computational Linguistics. https://doi.org/10.18653/v1/W18-0548

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Jain, S., Khangarot, H., & Singh, S. (2019). Journal recommendation system using content-based filtering. In J. Kalita, V. E. Balas, S. Borah, & R. Pradhan (Eds.), *Recent developments in machine learning and data analytics* (pp. 99–108). Springer. https://doi.org/10.1007/978-981-13-1280-9_9

Jelodar, H., Wang, Y., Yuan, C., Feng, X., Jiang, X., Li, Y., & Zhao, L. (2019). Latent Dirichlet allocation (LDA) and topic modeling: Models, applications, a survey. *Multimedia Tools and Applications*, *78*(11), 15169–15211. https://doi.org/10.1007/s11042-018-6894-4

Kamienski, A., Hindle, A., & Bezemer, C.-P. (2023). Analyzing techniques for duplicate question detection on Q&A websites for game developers. *Empirical Software Engineering*, *28*(1), 17. https://doi.org/10.1007/s10664-022-10256-w

Kanwal, S., Nawaz, S., Malik, M. K., & Nawaz, Z. (2021). A review of text-based recommendation systems. *IEEE Access*, *9*, 31638–31661. https://doi.org/10.1109/ACCESS.2021.3059312

Khashabi, D., Min, S., Khot, T., Sabharwal, A., Tafjord, O., Clark, P., & Hajishirzi, H. (2020, November). UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics* (EMNLP 2020), 16–20 November 2020, online (pp. 1896–1907). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.findings-emnlp.171

Kumar, A. P., Nayak, A., Ghosh, K., et al. (2023). A novel framework for the generation of multiple choice question stems using semantic and machine-learning techniques. *International Journal of Artificial Intelligence in Education*, 1–44. https://doi.org/10.1007/s40593-023-00333-6

Li, Y., Yao, L., Du, N., Gao, J., Li, Q., Meng, C., Zhang, C., & Fan, W. (2018). Finding similar medical questions from question answering websites. *arXiv preprint: 1810.05983*. https://doi.org/10.48550/arXiv.1810.05983

Long, P., Siemens, G., Conole, G., & Gasevic, D. (Eds.). (2011). *Proceedings of the First International Conference on Learning Analytics and Knowledge* (LAK 2011), 27 February–1 March 1, 2011, Banff, Alberta, Canada. ACM. https://doi.org/10.1145/2090116

Martinez-Gil, J., Freudenthaler, B., & Tjoa, A. M. (2019). Multiple choice question answering in the legal domain using reinforced co-occurrence. In S. Hartmann, J. Küng, S. Chakravarthy, G. Anderst-Kotsis, A. M. Tjoa, & I. Khalil (Eds.), *Proceedings of the 30th International Conference on Database and Expert Systems Applications* (DEXA 2019), 26–29 August 2019, Linz, Austria (pp. 138–148). Springer. https://doi.org/10.1007/978-3-030-27615-7_10

Mia, M. R., & Latiful Hoque, A. S. M. (2019). Question bank similarity searching system (QB3S) using LP and information retrieval technique. In *Proceedings of the First International Conference on Advances in Science, Engineering and Robotics Technology* (ICASERT 2019), 3–5 May 2019, Dhaka, Bangladesh (pp. 1–7). https://doi.org/10.1109/ICASERT.2019.8934449

Mitkov, R., Le An, H., & Karamanis, N. (2006). A computer-aided environment for generating multiple-choice test items. *Natural Language Engineering*, *12*(2), 177–194. https://doi.org/10.1017/S1351324906004177

Mudgal, S., Li, H., Rekatsinas, T., Doan, A., Park, Y., Krishnan, G., Deep, R., Arcaute, E., & Raghavendra, V. (2018). Deep learning for entity matching: A design space exploration. In *Proceedings of the 2018 International Conference on Management of Data* (SIGMOD 2018), 10–15 June 2018, Houston, Texas, USA (pp. 19–34). ACM. https://doi.org/10.1145/3183713.3196926

Mukherjee, S., & Kumar, N. S. (2019). Duplicate question management and answer verification system. In *Proceedings of the IEEE 10th International Conference on Technology for Education* (T4E 2019), 9–11 December 2019, Goa, India (pp. 266–267). https://doi.org/10.1109/T4E.2019.00067

Papasalouros, A., Kanaris, K., & Kotis, K. (2008). Automatic generation of multiple choice questions from domain ontologies. In *IADIS International Conference on e-Learning 2008*, 22–25 July 2008, Amsterdam, Netherlands (pp. 427–434, Vol. 1). IADIS. https://www.iadisportal.org/digital-library/automatic-generation-of-multiple-choice-questions-from-domain-ontologies

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830. https://www.jmlr.org/papers/v12/pedregosa11a.html

Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 3–7 November 2019, Hong Kong, China. Association for Computational Linguistics. https://doi.org/10.48550/arXiv.1908.10084

Reimers, N., & Gurevych, I. (2020). Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing* (EMNLP 2020), 16–20 November 2020, online (pp. 4512–4525). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-main.365

Salloum, S. A., Khan, R., & Shaalan, K. (2020). A survey of semantic analysis approaches. In A.-E. Hassanien, A. T. Azar, T. Gaber, D. Oliva, & F. M. Tolba (Eds.), *Proceedings of the International Conference on Artificial Intelligence and Computer Vision* (AICV 2020), 8–10 April 2020, Cairo, Egypt (pp. 61–70). Springer. https://doi.org/10.1007/978-3-030-44289-7_6

Samarinas, C., & Zafeiriou, S. (2019). Personalized high quality news recommendations using word embeddings and text classification models. *EasyChair Preprint*, *1254*, 2019.

Song, K., Tan, X., Qin, T., Lu, J., & Liu, T.-Y. (2020). Mpnet: Masked and permuted pre-training for language understanding. In *Proceedings of the 34th Conference on Advances in Neural Information Processing Systems* (NeurIPS 2020), 6–12 December 2020, online (pp. 16857–16867, Vol. 33). https://proceedings.neurips.cc/paper_files/paper/2020/hash/c3a690be93aa602ee2dc0ccab5b7b67e-Abstract.html

Tarrant, M., & Ware, J. (2012). A framework for improving the quality of multiple-choice assessments. *Nurse Educator*, *37*(3), 98–104. https://doi.org/10.1097/NNE.0b013e31825041d0

Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., & Zhou, M. (2020). MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *arXiv preprint: 2002.10957*. https://doi.org/10.48550/arXiv.2002.10957

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., . . . Rush, A. (2020). Transformers: State-of-the-art natural language processing. In Q. Liu & D. Schlangen (Eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations* (EMNLP 2020), 16–20 November 2020, online (pp. 38–45). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-demos.6

Wood, T. J. (2009). The effect of reused questions on repeat examinees. *Advances in Health Sciences Education*, *14*, 465–473. https://doi.org/10.1007/s10459-008-9129-z

Yang, Y., Cer, D., Ahmad, A., Guo, M., Law, J., Constant, N., Abrego, G. H., Yuan, S., Tar, C., Sung, Y.-H., Strope, B., & Kurzweil, R. (2019). Multilingual universal sentence encoder for semantic retrieval. *arXiv preprint: 1907.04307*. https://doi.org/10.48550/arXiv.1907.04307