SimRE: A Requirements Similarity Tool for Software Product Lines

María-Isabel Limaylla-Lunarejo
Universidade da Coruña, CITIC, Fac. Informática, Database Lab
A Coruña, Spain
maria.limaylla@udc.es

Nelly Condori-Fernandez
CiTIUS, Universidade de Santiago de Compostela
Santiago de Compostela, Spain
n.condori.fernandez@usc.es

Miguel R. Luaces
Universidade da Coruña, CITIC, Fac. Informática, Database Lab
A Coruña, Spain
miguel.luaces@udc.es

Abstract

A Software Product Line (SPL) is a paradigm that effectively describes families of products based on reuse. Requirements engineering in this domain is a complex task, especially when new products are introduced. In this context, identifying similarities between new and existing requirements can help avoid additional effort and duplication. On the other hand, with the accelerated progress of deep learning in textual analysis, particularly through pre-trained models, several opportunities for semantic analysis have emerged, including semantic similarity. However, most of the research in this area has focused on the English language, with limited attention given to other languages, such as Spanish.

In this paper, we introduce a novel tool, SimRE, that helps SPL engineers automatically identify the similarity of requirements written in Spanish using multilingual pre-trained models. We conducted a benchmarking study to validate and compare the performance of pre-trained models within the tool. This analysis aimed to provide a better understanding of their effectiveness in the Spanish language, particularly in the context of a Geographic Information System (GIS) product line.

CCS Concepts

• **Software and its engineering → Requirements analysis**; **Software product lines**; • **Computing methodologies → Language resources**.

Keywords

Requirements Similarity, Pre-trained models, Software Product Line

ACM Reference Format:
María-Isabel Limaylla-Lunarejo, Nelly Condori-Fernandez, and Miguel R. Luaces. 2025. SimRE: A Requirements Similarity Tool for Software Product Lines. In *The 40th ACM/SIGAPP Symposium on Applied Computing (SAC '25), March 31-April 4, 2025, Catania, Italy*. ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3672608.3707758

SAC '25, March 31-April 4, 2025, Catania, Italy

ACM ISBN 979-8-4007-0629-5/25/03
https://doi.org/10.1145/3672608.3707758

1 Introduction

A SPL is a family of software-intensive systems that share a common set of features to meet the specific needs of a particular segment [32]. The main advantage of an SPL is the reuse of common assets across the family of products. Given the complexity and variability of requirements in an SPL, having an effective requirements engineering (RE) process is crucial. RE is responsible for managing the requirements, including identifying if new requirements share features with existing ones and validating if a new product belongs to a specific product family. Identifying similarities between new and existing requirements can enhance the implementation of new products within the SPL paradigm.

In recent years, there have been significant advances in Natural Language Processing (NLP) tasks. The emergence of Transformer architecture [46] has been fundamental to the development of pre-trained and large language models (LLMs) [17]. These models, trained on a large corpus of text and with millions of parameters, have demonstrated promising results across several tasks, such as text classification, text generation, summarization, and sentence similarity. Several multilingual models are accessible from public repositories and have been applied to requirements similarity in recent studies [6, 16]. However, their application to SPL requirements management, particularly for similarity tasks, remains limited. A requirements similarity tool specifically oriented for SPL development would be highly beneficial given the variability of requirements within this paradigm. Integrating advances in NLP, particularly pre-trained models, such a tool can offer automated similarity assessments, significantly saving stakeholders and developers' time and promoting reuse efforts.

This study introduces SimRE, a Python-based tool designed to support reuse efforts in SPL development by utilizing several pre-trained models to calculate similarity scores between requirements. We conducted a benchmarking study to explore the effectiveness of these models, focusing on both accuracy and processing times when identifying similarities between requirements in the GIS/SPL domain. Our study also emphasizes requirements written in Spanish, a language of increasing importance as it is the fourth most spoken language globally and ranks second, after English, in the publication of scientific texts [23].

The main contributions of this paper are:

- A Python library for identifying requirements similarity in the SPL domain.
- A quantitative analysis of performance metrics comparing several pre-trained models for requirements similarities.
- An analysis of requirements similarity in the Spanish language for the GIS/LPS domain.

The paper has been organized in the following way. Section 2 presents some related works and Section 3 explains the tool. Section 4 detailed the benchmarking study and Section 5 presents the discussion of the results. Section 6 shows a usage scenario of our tool, and finally, Section 7 presents the conclusions and future work.

2 Related Work

Several studies have used various NLP techniques in the similarity process of requirements. A requirements similarity detection tool was proposed in [31]. This tool integrates several similarity detection algorithms, including the BM25F algorithm, which is based on TF-IDF and Jaccard similarity, and the FESVM algorithm, which combines feature extraction with a Support Vector Machine. Similarity detection using TF-IDF and cosine similarity was also proposed in [37]. Abbas et al. [2] conducted a comprehensive comparison of various text vectorization techniques to evaluate the similarity among requirements.

To improve reusability, ReSim, a framework for identifying similarity between requirements, was presented in [18]. The semantic-based hybrid model used (based on WordNet and vector similarity at word level) outperformed Dice, Jaccard, and Cosine techniques. Das et al. introduced two BERT-based models PUBER and FiBER [16]. FiBER, fine-tuned with the PURE dataset, possesses a vast English vocabulary and comprehends words from the RE domain. In contrast, PUBER, trained exclusively on the PURE requirements dataset, includes words from the RE domain but lacks the extensive English vocabulary of BERT. An experiment comparing several techniques, like traditional (JSI and TFIDF), word2vec-based and BERT, Sentence-BERT (ST-Roberta and Mini-LM) and USE, was performed in [1]. The sentence BERT obtained the best f1-score over the others models (0.84), with USE very close.

Regarding the SPL domain, techniques such as WordNet, POS tagging, Named Entity Recognition (NER), and DISCO were used to expand semantic information and facilitate finding similarities in the current functionality [7]. Bakar used Latent Semantic Analysis (LSA) and Vector Space Model for similar requirements identification and clustering techniques for feature extractions from requirements and reuse [8]. Abbas et al. [3] employed some NLP tasks to clean the requirements (stopwords, lemmatization, POS tagging), then predicted clusters for vectors (using word embeddings and k-means algorithms) on existing clusters, and finally obtained the closest requirements. In [9], a transformation of requirements into a semantic tree model was performed using several NLP techniques, such as tokenization, POS, and chunking. The hierarchical semantic representation of requirements is constructed with a taxonomy of SPL business domain to facilitate the similarity process. Finally, a J48 algorithm is used to retrieve commonalities and variabilities to obtain similar rankings based on requirements in [20].

We found that, although NLP techniques have been employed in various studies to identify similarities among requirements, the application of pre-trained models for this task in the context of Software Product Lines (SPL) is still limited.

3 SimRE

We developed a tool, called SimRE, with the capacity to perform similarity detection using a combination of all models, but also allowing users to choose the models. Our tool enables users to automatically detect similarities between new requirements and existing requirements/features, using a similarity process explained in subsection 3.1.

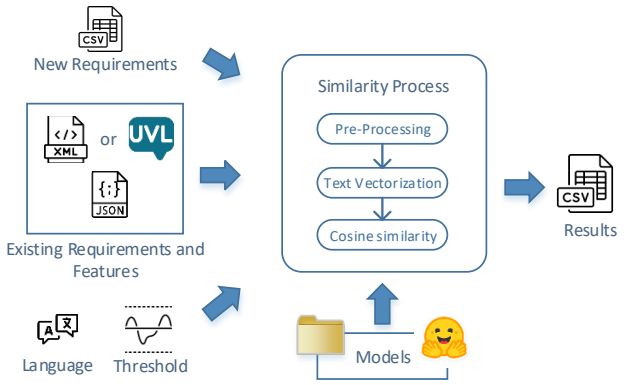

Figure 1: Tools' Design

The tool not only shows the requirements with the highest similarity (according to the threshold), but also highlights the features associated with these requirements. Features in the SPL domain are crucial as they are easy to identify in the feature model, and their hierarchical relationships are easy to understand.

A general view of the design is shown in Figure 1. SimRE receives both new requirements (as a CSV file) and existing requirements (as a feature model and a JSON file describing the features), and then performs a similarity process between each requirement. After processing, a list of existing requirements is provided, each assigned a similarity score that shows its degree of similarity to the new requirements. In addition to a list of new requirements (which may originate from new projects or products), the feature model and its requirements are also the main input to the tool. SimRE was developed to support two formats for feature models: FeatureIDE, an open-source framework for feature-oriented software development (based on Eclipse) [44], and UVL (Universal Variability Language), a textual format for variability modeling [43]. Figure 2 shows examples of both formats.

The feature model must be associated with a JSON file that contains the descriptions of requirements belonging to the corresponding feature, as shown in Figure 3. The tool also supports two languages: Spanish and English. New requirements to be analyzed should be provided as a list in a CSV file.

Additionally, the tool can receive three other optional inputs. The first one is a threshold that sets the minimum similarity score required to filter results. Only existing requirements with a similarity score above this threshold will be included in the results. If more than one model is chosen, this value will be the average of all models. The second input is the models. One model is selected by default, but the tool can use a combination of five models (subsection 3.2) or use each of the models separately. Since the tool uses pre-trained models, the processing time increases with each additional model. The last option is a pre-processing setting, indicating whether the tool should perform the pre-processing step.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<featureModel chosenLayoutAlgorithm="4">
   <struct>
      <and mandatory="true" name="GEMA_SPL">
         <and name="UserManagement">
            <or name="UM_Registration">
               <feature mandatory="true" name="UM_R_ByAdmin"/>
               <feature mandatory="true" name="UM_R_Anonymous"/>
   </struct>
</featureModel>
```

(a)

```
features
        UserManagement
                optional
                        UM_Registration
                                or
                                        UM_R_ByAdmin
                                        UM_R_Anonymous
```

(b)

Figure 2: Format's input files. (a) FeatureIDE XML format and (b) UVL format

```
{
  "MV_LM_HideLayer": {
  "desc": "Enables you to select whether you want to show or hide each
           layer of the map."  },
  "UM_Authentication": {
  "desc": "User Authentication" }
}
```

Figure 3: JSON format for requirement's description

The tool was developed using the Python language, chosen for its many libraries for managing transformer-based models. The source code is available on a public GitHub repository[1]. The code is written in Python 3.8.10. In addition, to enhance the user experience, a simple web user interface based on Django was also developed[2].

3.1 Similarity process

The similarity process consists of three steps: The first step is *pre-processing step*, which is performed on both sets of requirements. In this step, each requirement is tokenized using the SpaCy library[3] and its Spanish language pipeline. SpaCy supports more than 70 languages. Additionally, stopwords are removed during tokenization. The second step is *Text vectorization* through embedding. This can be performed using some models and their different methods, explained in detail in subsection 3.2. The last step is the calculation of the distance between two sentence embeddings using the *cosine similarity* measure. This score, ranging from zero to one, represents the degree of similarity between the sentences: a value closer to one indicates greater similarity, and a value closer to zero indicates less similarity. Cosine similarity is a term-based technique that has been used for text similarity approaches [19].

[1]https://github.com/lbdudc/simre
[2]https://github.com/lbdudc/simre-ui
[3]https://spacy.io/

3.2 Pre-trained Models

Due to significant advancements in pre-trained models across various natural language processing (NLP) tasks, numerous models are available in diverse repositories, accessible for public utilization. *Transformers* is an open-source library designed to facilitate user access to large-scale pre-trained models, allowing them to build and experiment on top of these models, and to deploy them in downstream tasks with state-of-the-art performance [35]. Hugging face[4] is a repository where several models are submitted by major companies and individual users.

The pre-trained models utilized in this work are the following:

A. Word2vec

Word2vec is a word embedding technique presented by Mikolov et al [30]. We used the Spanish Billion Words Corpus and Embeddings (SBWCE) model (SBW-vectors-300-min5.bin) [10]. We employed the gensim library [38] to manage these models.

B. fastText

FastText [25] is an open-source library released in 2016 by Facebook Research that provided word vectors for 157 languages. These word vectors were trained using the Continuous Bag of Words (CBOW) on Common Crawl and Wikipedia. We used the word vectors for Spanish that can be downloaded from the FastText web page[5].

C. Sentence BERT-based

Reimers and Gurevych (2019) introduced Sentence-BERT (SBERT), a modification of the BERT architecture to perform sentence embedding [39]. In 2020 they presented a method to make monolingual Sentence Embedding Multilingual, generating several models in more than 50 languages, including Spanish [40]. Three multilingual models, obtained from HuggingFace, a repository of models, datasets, and more, were used for sentence embedding. We used the Sentence Transformers library, a Python framework for sentence embedding based on the transformers architecture, to manage these models. The models used are the following:

- *MiniLM-L12-v2*[6]: Multilingual version of paraphrase-MiniLM-L12-v2, a model that maps sentences to a 384-dimensional dense vector space.
- *mpnet-base-v2*[7]: Multilingual version of paraphrase-mpnet-base-v2, a model that maps sentences to a 768-dimensional dense vector space.
- *distiluse-cased-v2*[8]: A distilled version of the universal sentence encoder (USE).

According to [40], the main datasets used for training sentence-based models include WikiMatrix (parallel sentences from Wikipedia in various languages), Tatoeba, Europarl, NewsCommentary, GlobalVoices, JW300 (mined sentences from the magazines *Awake!* and *Watchtower*), and translated movie subtitles from OpenSubtitles.org and from approximately 4,000 TED talks.

[4]https://huggingface.co
[5]https://fasttext.cc/docs/en/crawl-vectors.html
[6]t.ly/AWU1a
[7]t.ly/7Fg1p
[8]t.ly/6k-90

4 Experiment and Results

This section presents the research questions, a description of the dataset used, and an overview of the results of the benchmarking study performed to compare and validate the models integrated into the tool.

4.1 Research Method

We conducted a benchmarking study to compare the performance of several pre-trained models on a requirements similarity task. We focused on GIS/SPL domain and the Spanish language. We also aim to obtain a Requirement Pair dataset with requirements based on the GIS domain to investigate the effectiveness of these models in this domain. From this goal, the following research questions were formulated:

- **RQ1**: How do several multilingual pre-trained models perform in terms of accuracy and processing time when measuring the similarity between software requirements in the context of GIS/SPL?
- **RQ2**: What are the differences in the effectiveness of these pre-trained models when applied to varying descriptions of software requirements?
- **RQ3**: Is it useful to combine several pre-trained models to improve performance in identifying GIS/SPL requirement similarity?

To address RQ1, we calculated the similarity scores using the five models on the second dataset and compared their performance metrics, specifically accuracy and processing time. For RQ2, we enhanced the descriptions of the requirements in the first dataset. Finally, for RQ3, we explored different methods of combining the models' results.

From our research questions (RQs), we identified the *similarity metrics* as our dependent variable, and the requirements in the datasets and the models as the independent variable. We employed Pearson Correlation [29] and Spearman Rank Correlation [42] as evaluation metrics to assess the relationship between human-annotated scores and the similarity scores generated by the models. Pearson correlation, which measures the linear relationship between two variables, and Spearman correlation, which captures the relationship between variables regardless of linearity, have been widely utilized in several studies on text similarity [1, 2, 6, 12]. Their relevance has also been mentioned in literature reviews by Prakoso et al. [36] and He et al. [22].

4.2 Datasets

Two datasets were used in this experiment: the GIS dataset and the RequirementPairs dataset.

A. GIS Dataset

The GIS dataset is composed of the requirements of an Software Product Line (SPL) project [15], containing 173 requirements written in both Spanish and English concerning GIS. These requirements represent the functionality of a web-based GIS product line, with each requirement associated with specific features. The requirement descriptions are in JSON format, associated with the corresponding features. We employed two versions of this dataset: the first version (V1) contained concise requirement descriptions, while the second version (V2) provided more detailed descriptions. Table 1 presents some statistics comparing both versions, including word counts and average for requirements, and counts of specific Part-of-Speech (POS) tags such as verbs (V), nouns (N), and adjectives (Adj.). Additionally, it presents POS interactions for SBERT models as referenced in [45]. Both versions have been shared in the public repository Zenodo[9].

Table 1: Statistic of GIS's dataset (V1/V2)

	Word avg.	Word count	Distribution of main POS Tags						
			V	Adj	N	N-N	N-V	V-V	N-Adj
V1	8	1372	215	90	378	6	4	60	59
V2	27	4755	527	328	1370	10	21	39	243

B. RequirementPairs Dataset

We observed a lack of labeled datasets for similarity tasks in the Spanish language compared to the abundance available for English. For instance, [13] lists at least 17 benchmark datasets in English from 1965 to 2019. One example for the Spanish language is the SemEval Spanish STS dataset [5, 11], which was utilized in a task at the SemEval Workshop since 2014. To help bridge this gap, we created a small labeled dataset specifically focused on the GIS domain, aimed at validating similarity between requirements in this context.

This dataset initially consists of 30 requirements and was prepared by the Laboratory of Database from the University of A Coruña. The dataset was created using the following steps: first, 10 requirements were extracted from the PROMISE dataset [41], which had been previously translated into Spanish in previous research[10]. We selected ten requirements unrelated to the GIS domain, with the expectation that no significant similarities would be found. Second, the first author generated an additional 10 requirements based on various GIS requirements. Finally, the remaining 10 requirements were created using ChatGPT with the following prompt in Spanish.

"*Considering the following definitions: Software Requirement: A need identified by a stakeholder, or a capability or property that a system must have. Geographic Information System (GIS): GIS is a technology designed to capture, store, manipulate, analyze, manage, and present all types of geographical data. In simpler terms, a GIS is a tool that allows users to create interactive queries (search for geographical and visual information), analyze spatial information, and edit data, maps, and presentations. Generate 10 software requirements for a GIS system in Spanish.*"

Manual labeling was conducted by two native Spanish speakers to identify similarities between these 30 requirements and the GIS requirements and features, and to determine the degree of these similarities. The participants (annotators), who belong to the university's laboratory and have extensive knowledge of the GIS domain, agreed to participate in this study. Each annotator received an Excel file containing the 30 requirements, along with a detailed description of the study's objectives and instructions.

[9] https://doi.org/10.5281/zenodo.14506521
[10] https://doi.org/10.5281/zenodo.7311148

Table 2: RequirementPairs Dataset

Id	Requirement	GIS Dataset		Similarity Degree
		Feature	Requirement	
4	The product should allow exporting maps in various formats, either as images or PDFs.	MV_T_Export	Add the functionality to export the map in various formats. Include a control on the map that allows selecting the export format and downloading it.	5
4	The product should allow exporting maps in various formats, either as images or PDFs.	MV_T_E_F_PNG	Allow export to PNG format.	5
7	The system will allow customers to pay for a streaming movie with a prepaid card.	T_P_RedSys	Add a component that enables payments via RedSys.	4
15	The system will automatically generate an inventory quantity adjustment document when the daily product sales data becomes available.	T_Quartz	It allows enabling the Quartz Scheduler to perform periodic tasks on the server.	2

The GIS dataset, including the features and their corresponding requirement descriptions, was also provided.

The annotators were asked to associate up to five GIS features or requirements with each given requirement based on similarity, assigning each a rating from one to five, where one indicates low similarity and five indicates high similarity. To validate the agreement of the results, we used the kappa coefficient, proposed by Cohen [14] to calculate the reliability between the two annotators. This coefficient is one of the standard ways to measure meaningful agreement between annotators and has been used on several studies when generating a training dataset associated to software requirements [4, 24, 27, 33]. According to the level of agreement of Landis and Koch [28], we obtained a Substantial Agreement (0.69) on Cohen's kappa statistic based only on features, and a Moderate Agreement (0.45) when combine the features with the similarity degree, considering only the matches found in the results for each requirement. Table 3 shows the annotators' results categorized by the origin of the requirements. We identified 14 requirements that correspond to the same GIS feature with an identical similarity rating of five. Seven requirements share the same feature, but differ in their similarity ratings. Finally, nine requirements showed no overlap in similar features between the annotators

Table 3: Annotators Results

Source	Same feature and similarity degree	Same feature, not similarity degree	Without features similarities
PROMISE	2	2	6
CHATGPT	3	4	3
First author	9	1	0
Total	**14**	**7**	**9**

To create the final version of this dataset, we incorporated the results from the initial 14 requirements, as well as the seven that shared features but had differing similarity ratings. Additionally, we included other features that were common between both annotators. This resulted in 34 combinations of the 21 requirements with the features/requirements from the GIS dataset, along with their corresponding similarity ratings. In cases where the ratings differed, we selected the higher value. An example is shown in Table 2. We have shared this dataset in Zenodo[11].

[11]https://doi.org/10.5281/zenodo.14506521

4.3 Data analysis and Results

A. Comparison of pre-trained model for similarity measurement (RQ1)

We performed the benchmarking study and obtained the results using the first version of the GIS dataset (V1 on Table 4). The processing time is the time required for the model to obtain the vectorization and cosine similarity for a new single requirement compared with all the GIS dataset. It is shown in the format mm:ss.ms (minutes:second.milliseconds). The time needed for the model to upload is not considered in this metric. Pearson's and Spearman rank correlation coefficients were employed as the evaluation metric to compare the performance of the models. We used the function corr of the pandas library, which implements these metrics.

The Sentence BERT-base models exhibited the longest processing time for a single requirement, with the 'paraphrase-multilingual-mpnet-base-v2' model taking the longest time, approximately 25 seconds and 5 milliseconds. However, this model outperformed other models in terms of Pearson and Spearman correlation, achieving the highest value of 0.66 and 0.63. The 'paraphrase-multilingual-MiniLM-L12-v2' model achieved the second-best performance. In contrast, the 'distiluse-base-multilingual-cased-v2' model had the lowest Pearson correlation value. While Word2Vec and fastText algorithms were the fastest, their correlation scores were not the highest.

Table 4: Processing time and performances of the Pearson/Spearman correlation (RQ1/RQ2)

Models	V1			V2		
	Time	Pearson	Spearman	Time	Pearson	Spearman
MiniLM-L12-v2	00:13.32	0.60	0.62	00:13.63	**0.62**	0.52
distiluse-cased-v2	00:15.68	0.44	0.43	00:15.36	0.55	**0.58**
mpnet-base-v2	00:25.05	**0.66**	**0.63**	00:25.73	0.57	0.51
word2vec	00:02.52	0.53	0.57	00:02.62	0.61	0.54
fastText	00:03.74	0.54	0.45	00:04.88	0.52	0.41

B. Comparison when applied a varying description of GIS dataset (RQ2)

We enhanced the descriptions of the requirements in the GIS dataset by providing more detailed explanations, which included additional words to better clarify the features. We measured both the processing time and Pearson and Spearman correlation for these descriptions, and the results are presented in Table 4 (V2).

The processing times follow the same pattern as in the previous case, with Sentence BERT-based models having the highest processing times and Word2Vec and fastText algorithms exhibiting the lowest. No significant difference in processing times was observed between the two versions of the GIS dataset.

In terms of Pearson's correlation, the 'paraphrase-multilingual-MiniLM-L12-v2' model outperformed the others, with Word2Vec following in second place. Spearman rank correlation indicates that the 'distiluse-base-multilingual-cased-v2' model outperformed the other models. FastText had the lowest correlation value among the models for both correlation metrics.

C. Comparison when combining several models (RQ3) In this case, we calculated metrics for several model combinations. Table 5 presents the metrics for both versions of the GIS dataset across several combinations. These metrics were calculated based on the average similarity degree of the models in the corresponding combination.. In the analysis of GIS dataset V1, the results indicate that the combination of model 3 (paraphrase-multilingual-mpnet-base-v2) with models 4 (word2vec), or 5 (fastText), or both achieved higher performance compared to using model 3 alone, according to Pearson (0.70 compared to 0.66) and Spearman correlation (0.64 compared to 0.63). The processing time in these combinations also increased a little more than using only the model 3.

Similarly, in GIS dataset V2, the combination of model 1 (paraphrase-multilingual-MiniLM-L12-v2) with model 4, as well as the combination of models 1, 4, and 5, achieved better results compared to model 1 alone according to Pearson correlation, increasing from 0.62 to 0.68. Combining model 2, which achieved the highest Spearman correlation, with model 4 resulted in a slight improvement in performance (from 0.58 to 0.60).

Table 5: Processing time and performances of the Pearson/Spearman correlation for each version of GIS dataset(RQ3)

Combinations	V1			V2		
	Time	Pearson	Spearman	Time	Pearson	Spearman
All models	00:59.41	0.63	0.59	01:00.78	0.64	0.58
model 1, 2 and 3	00:59.78	0.60	0.62	00:43.72	0.61	0.57
model 1 and 2	00:29.65	0.56	0.57	00:23.58	0.61	0.59
model 1 and 3	00:39.09	0.64	0.63	00:33.03	0.61	0.59
model 2 and 3	00:38.11	0.58	0.57	00:34.39	0.59	0.57
model 4 and 5	00:10.89	0.56	0.51	00:07.55	0.58	0.50
model 1 and 5	00:14.10	0.66	0.60	00:13.54	0.65	0.54
model 2 and 5	00:18.98	0.51	0.43	00:16.58	0.60	0.57
model 3 and 5	00:28.09	**0.70**	0.62	00:23.85	0.60	0.52
model 1 and 4	00:16.45	0.63	0.63	00:12.66	**0.68**	0.57
model 2 and 4	00:18.45	0.50	0.48	00:15.92	0.62	**0.60**
model 3 and 4	00:26.87	**0.67**	**0.64**	00:23.29	0.64	0.52
model 1, 4 and 5	00:16.47	0.65	0.60	00:14.51	**0.68**	0.54
model 2, 4 and 5	00:18.68	0.54	0.49	00:17.79	0.63	0.57
model 3, 4 and 5	00:33.58	**0.68**	**0.64**	00:24.57	0.64	0.51

Legend: model 1:Multilingual MiniLM-L12-v2, model 2:Multilingual distiluse-cased-v2, model 3:Multilingual mpnet-base-v2, model 4: word2vec, model 5: fastText

5 Discussion

We observed that sentence-based models outperform word2vec and fastText (RQ1). This result is consistent with other research showing that BERT-based models achieve higher performance in similarity tasks compared to word2vec and fastText [21, 34]. The advantage of sentence-based models lies in their training, which includes not only word-level understanding but also the context of entire sentences. This finding supports the work of other studies [1, 12] that have demonstrated better performance of sentence-based models in text similarity tasks. However, one major disadvantage is their processing time, particularly if they are utilized for real-time use. Future work should explore implementing these models as asynchronous tasks to efficiently manage larger datasets and reduce delays.

Furthermore, the results indicate that the Pearson and Spearman correlation values are relatively modest (not exceeding 0.66). This could be attributed to two key factors: first, the models are trained on general corpora, whereas our dataset is focused on a specialized domain. The corpus used for training these pre-trained models covers a broad range of topics, including politics and economics, but lack specialization and, in some cases, may not be entirely accurate. Second, multilingual models could exhibit weaknesses across different languages (in this case, Spanish). For example, in [26], Sentence BERT was tested across multiple languages for Semantic Textual Relatedness (STR), which measures the degree of semantic similarity between pairs of sentences. Their results show that multilingual models, especially those based on English, outperformed the Spanish models and other low-resource languages.

We also validated the impact of varying the description of the base requirements (GIS dataset) (RQ2). The results show that even when we improved the requirements of GIS dataset, the metrics obtained different values than the previous version: Depending on the metrics and the model some values increased while other decreased. Since there is limited research explaining the causes of BERT-based models' predictions, the reasons for these differences may be attributed to several factors. For instance, Vasileiou and Eberle examine Transformer-based similarity models and presented a corpus-level analysis [45]. Following this, we found that Version two includes a greater quantity of the primary POS tags, except for Verb-Verb combinations (e.g., 'Enables centering'). This observation, however, does not explain for the observed differences. Further investigation is required to explain this case.

A combination of several models was also performed to validate the performance (RQ3). The results indicate an improvement in performance, as measured by Pearson and Spearman correlation, when combining various models. The performance increases when a Sentence BERT-based model is combined with either fastText or word2vec. Our analysis revealed that the optimal combination of models differs between the two versions of the GIS dataset. Therefore, we aim to provide users with the flexibility to select a model or a combination of models based on their needs, while also considering the processing time associated with each option.

5.1 Threats to Validity

Threats to *internal validity* mainly concern the quality of the requirements in the dataset for our benchmarking study. To obtain reliable data on the performance of requirements similarity, it is crucial to select the test requirements accurately. To mitigate author bias in creating RequirementsPair dataset, we used requirements from the PROMISE benchmark dataset and some generated by ChatGPT. Two annotators also labeled these requirements. An increase in the

number of annotators is necessary for future work, as the level of agreement between annotators may vary. Concerning threats to *external validity*, although our benchmarking study focuses on GIS domain and is based on real-world development projects, it is possible that some aspects were not considered in our experiment. For future work, incorporating a broader range of GIS requirements and testing across additional domains within the SPL paradigm will be essential to enhance generalisation. We were also concerned with the *construct validity* of the similarity metrics. To ensure robust comparisons, we selected cosine similarity, a well-known metric, to evaluate performance.

6 A Usage Scenario

This section describes a scenario to demonstrate how to use the tool. The tool can be used in two ways: using *pip*[12] or through an interface by Django. Before executing the tool, several libraries and models need to be installed. Detailed instructions for technical implementation and the commands to run the tool are provided in the project's README file. An example of the web interface is shown in Figure 4.

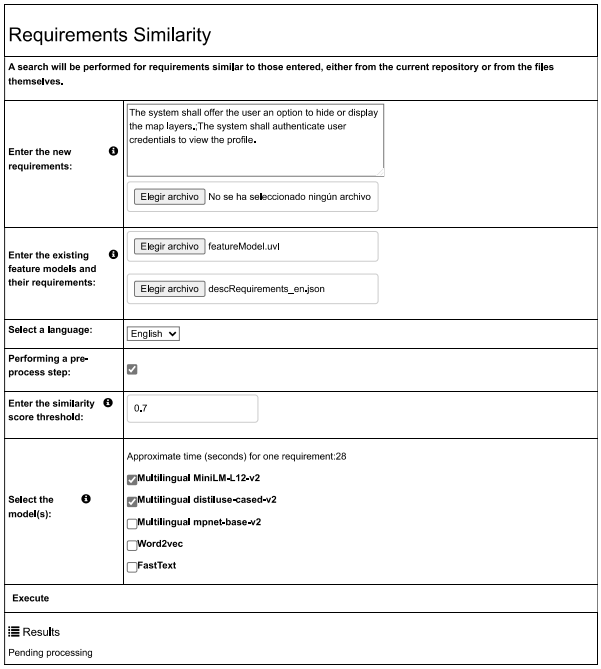

Figure 4: SimRE Web Interface

The model 'MiniLM-L12-v2' is selected as the default option due to its balance between execution time and performance metrics. Although it is not the slowest among all models, it provides the second-best Pearson and Spearman correlation values. The tool also provides estimated completion times for each checked model, allowing users to know the processing duration for one requirement. The default threshold value is set to 0.7 for English sentences. However, for processing requirements in Spanish, we recommend adjusting the threshold to 0.6 or 0.5 to optimize results.

[12]https://pypi.org/project/simre/

We tested the following two new requirements:

"*R1: The system shall offer the user an option to hide or display the map layers.*"

"*R2: The system shall authenticate user credentials to view the profile.*" The result provides the following data:

Results of the similarity process are presented on CSV format. Table 6 shows the main results obtained for this usage scenario. For Requirement 1 (R1), the tool shows two options with scores of 0.778 and 0.731. The first option closely matches the requirement, while the second option has similarities but is more generalized. Requirement 2 (R2) has one result with a score of 0.716.

Table 6: Results

	Score	Similar Requirement	Feature
R1	0.778	Enables you to select whether you want to show or hide each layer of the map.	MV_LM_HideLayer
R1	0.731	Enables to manage the layers displayed on the map.	MV_LayerManagement
R2	0.716	User Authentication	UM_Authentication

Finally, SimRE presents several advantages. Although most of the pre-trained models are accessible through Hugging Face and others repositories, our tool simplifies access to these models and a combination of them. Moreover, it shows an estimation of time based on the combination of selected models. It also provides the option to set a threshold so only requirements from the repository with a similarity score above the specified threshold are showed.

7 Conclusions

This paper introduces SimRE, a Python-based tool for automatically identifying requirement similarities in SPL projects. SimRE uses several pre-trained models in both Spanish and English, integrating them into a user-friendly tool to enhance requirements reuse. Our tool also provides users with the flexibility to select and combine these models, depending on whether their priority is improved performance or faster processing times.

We also conducted a benchmarking study to evaluate the effectiveness of the pre-trained models integrated into the tool for analyzing a GIS/LPS domain case with requirements written in Spanish. The results show that BERT-based sentence models outperformed the other models, despite their longer processing time. Moreover, findings show that a combination of these models with word2vec or even fastText could improve the performance. Despite the modest performance results, these models demonstrate potential for identifying requirement similarities in Spanish.

In future work, we plan to perform evaluations on real-life projects to validate the effectiveness of the tool in other domains. Furthermore, several improvements can be made to the tool: integrating LLMs such as GPT or LLaMA, supporting various feature model formats, and developing a web-based tool for data persistence and requirements management.

Acknowledgments

This research was partially supported by Xunta de Galicia/FEDER-UE ED413C 2021/53 (Database Lab, UDC) and Galician Ministry of Culture, Education, Professional Training, and University (grants

ED431G2019/04, ED431C2022/19). All grants were co-funded by the European Regional Development Fund (ERDF/FEDER program).

References

[1] Muhammad Abbas, Sarmad Bashir, Mehrdad Saadatmand, Eduard Paul Enoiu, and Daniel Sundmark. 2024. Requirements Similarity and Retrieval. Springer. http://www.ipr.mdu.se/publications/6922- This preprint will appear as a chapter in a book provisionally titled "Handbook on Natural Language Processing for Requirements Engineering", to be published by Springer..

[2] Muhammad Abbas, Alessio Ferrari, Anas Shatnawi, Eduard Enoiu, Mehrdad Saadatmand, and Daniel Sundmark. 2023. On the relationship between similar requirements and similar software: A case study in the railway domain. *Requirements Engineering* 28, 1 (2023), 23–47.

[3] Muhammad Abbas, Mehrdad Saadatmand, Eduard Enoiu, Daniel Sundamark, and Claes Lindskog. 2020. Automated reuse recommendation of product line assets based on natural language requirements. In *International Conference on Software and Software Reuse*. Springer, 173–189.

[4] Sallam Abualhaija, F Basak Aydemir, Fabiano Dalpiaz, Davide Dell'Anna, Alessio Ferrari, Xavier Franch, and Davide Fucci. 2024. Replication in Requirements Engineering: The NLP for RE Case. *ACM Transactions on Software Engineering and Methodology* 33, 6 (2024), 1–33.

[5] Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez Agirre, Rada Mihalcea, German Rigau Claramunt, and Janyce Wiebe. 2016. Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *SemEval-2016. 10th International Workshop on Semantic Evaluation; 2016 Jun 16-17; San Diego, CA. Stroudsburg (PA): ACL; 2016. p. 497-511.* ACL (Association for Computational Linguistics).

[6] Meizan Arthur Alfianto, Yudi Priyadi, and Kusuma Ayu Laksitowening. 2023. Semantic Textual Similarity in Requirement Specification and Use Case Description based on Sentence Transformer Model. In *2023 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT)*. IEEE, 220–226.

[7] Maximiliano Arias, Agustina Buccella, and Alejandra Cechich. 2018. A framework for managing requirements of software product lines. *Electronic Notes in Theoretical Computer Science* 339 (2018), 5–20.

[8] Noor Hasrina Bakar. 2016. *Feature Extraction from Natural Language to Aid Requirements Reuse in Software Product Lines Engineering*. University of Malaya (Malaysia).

[9] Anissa Benlarabi, Amal Khtira, Bouchra El Asri, et al. 2020. Learning to Support Derivation of Adaptable Products in Software Product Lines. *Journal of Computer and Communications* 8, 04 (2020), 114.

[10] Cristian Cardellino. 2019. Spanish Billion Words Corpus and Embeddings. https://crscardellino.github.io/SBWCE/

[11] Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055* (2017).

[12] Dhivya Chandrasekaran and Vijay Mago. 2021. Comparative analysis of word embeddings in assessing semantic similarity of complex sentences. *IEEE Access* 9 (2021), 166395–166408.

[13] Dhivya Chandrasekaran and Vijay Mago. 2021. Evolution of semantic similarity—a survey. *ACM Computing Surveys (CSUR)* 54, 2 (2021), 1–37.

[14] Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement* 20, 1 (1960), 37–46.

[15] Alejandro Cortiñas, Miguel R. Luaces, Oscar Pedreira, Ángeles S. Places, and Jennifer Pérez. 2017. Web-based geographic information systems SPLE: Domain analysis and experience report. In *ACM International Conference Proceeding Series*, Vol. 1. https://doi.org/10.1145/3106195.3106222

[16] Souvick Das, Novarun Deb, Agostino Cortesi, and Nabendu Chaki. 2021. Sentence embedding models for similarity detection of software requirements. *SN Computer Science* 2 (2021), 1–11.

[17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[18] Ahmad Farooq and Mohammad Faisal. 2023. Assessing Similarity between Software Requirements: A Semantic Approach. *International Journal of Information Engineering and Electronic Business* 13, 2 (2023), 38.

[19] Wael H Gomaa, Aly A Fahmy, et al. 2013. A survey of text similarity approaches. *international journal of Computer Applications* 68, 13 (2013), 13–18.

[20] Wasi Haider, Yasir Hafeez, Sadia Ali, Azeem Abbas, M Numan Rafi, and Abdul Salam. 2019. Improving Requirement Prioritization process in Product line using Artificial Intelligence technique. *KIET Journal of Computing and Information Sciences* 2, 2 (2019), 12–12.

[21] Mengting Han, Xuan Zhang, Xin Yuan, Jiahao Jiang, Wei Yun, and Chen Gao. 2021. A survey on the techniques, applications, and performance of short text semantic similarity. *Concurrency and Computation: Practice and Experience* 33, 5 (2021), e5971.

[22] ZHENGFANG HE, CRISTINA E DUMDUMAYA, and VALA QUIMNO. 2024. MEASUREMENT OF SEMANTIC TEXT SIMILARITY. *Journal of Theoretical and Applied Information Technology* 102, 5 (2024).

[23] Instituto Cervantes. 2022. El Español una lengua viva. https://cvc.cervantes.es/lengua/espanol_lengua_viva/pdf/espanol_lengua_viva_2022.pdf [Online; accessed 23-April-2024].

[24] Tahira Iqbal, Moniba Khan, Kuldar Taveter, and Norbert Seyff. 2021. Mining reddit as a new source for software requirements. In *2021 IEEE 29th International Requirements Engineering Conference (RE)*. IEEE, 128–138.

[25] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759* (2016).

[26] Senthil Kumar, Aravindan Chandrabose, B Gokulakrishnan, and Karthikraja Tp. 2024. NLP_Team1@ SSN at SemEval-2024 Task 1: Impact of language models in Sentence-BERT for Semantic Textual Relatedness in Low-resource Languages. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*. 1854–1859.

[27] Zijad Kurtanović and Walid Maalej. 2017. Mining user rationale from software reviews. In *2017 IEEE 25th international requirements engineering conference (RE)*. IEEE, 61–70.

[28] J Richard Landis and Gary G Koch. 1977. The measurement of observer agreement for categorical data. *biometrics* (1977), 159–174.

[29] Joseph Lee Rodgers and W Alan Nicewander. 1988. Thirteen ways to look at the correlation coefficient. *The American Statistician* 42, 1 (1988), 59–66.

[30] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[31] Joaquim Motger de la Encarnación, Cristina Palomares Bonache, and Jordi Marco Gómez. 2020. RESim-Automated detection of duplicated requirements in software engineering projects. In *Joint Proceedings of REFSQ-2020 Workshops, Doctoral Symposium, Live Studies Track, and Poster Track: co-located with the 26th International Conference on Requirements Engineering: Foundation for Software Quality (REFSQ 2020): Pisa, Italy, March 24, 2020*. CEUR-WS. org, 1–6.

[32] Linda Northrop. 2005. *Software product lines*. Technical Report. CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.

[33] Olga Ormandjieva, Ishrar Hussain, and Leila Kosseim. 2007. Toward a text classification system for the quality assessment of software requirements written in natural language. In *Fourth international workshop on Software quality assurance: in conjunction with the 6th ESEC/FSE joint meeting*. 39–45.

[34] Avinash Patil, Kihwan Han, and Aryan Jadon. 2024. A Comparative Analysis of Text Embedding Models for Bug Report Semantic Similarity. In *2024 11th International Conference on Signal Processing and Integrated Networks (SPIN)*. IEEE, 262–267.

[35] Juan Manuel Pérez, Juan Carlos Giudici, and Franco Luque. 2021. pysentimiento: A Python Toolkit for Sentiment Analysis and SocialNLP tasks. *arXiv preprint arXiv:2106.09462* (2021).

[36] Dimas Wibisono Prakoso, Asad Abdi, and Chintan Amrit. 2021. Short text similarity measurement methods: a review. *Soft Computing* 25 (2021), 4699–4723.

[37] Sandeep Reddivari. 2021. T-ReQs: A tool for tracking similarity in software requirements. In *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE, 1409–1410.

[38] Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, 45–50. http://is.muni.cz/publication/884893/en.

[39] Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084* (2019).

[40] Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. *arXiv preprint arXiv:2004.09813* (2020).

[41] J. Sayyad Shirabad and T.J. Menzies. 2005. The PROMISE Repository of Software Engineering Databases. School of Information Technology and Engineering, University of Ottawa, Canada. http://promise.site.uottawa.ca/SERepository

[42] Charles Spearman. 1961. The proof and measurement of association between two things. (1961).

[43] Chico Sundermann, Kevin Feichtinger, Dominik Engelhardt, Rick Rabiser, and Thomas Thüm. 2021. Yet another textual variability language? a community effort towards a unified language. In *Proceedings of the 25th ACM International Systems and Software Product Line Conference-Volume A*. 136–147.

[44] Thomas Thüm, Christian Kästner, Fabian Benduhn, Jens Meinicke, Gunter Saake, and Thomas Leich. 2014. FeatureIDE: An extensible framework for feature-oriented software development. *Science of Computer Programming* 79 (2014), 70–85.

[45] Alexandros Vasileiou and Oliver Eberle. 2024. Explaining Text Similarity in Transformer Models. *arXiv preprint arXiv:2405.06604* (2024).

[46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).