

Requirements classification using fastText and BETO in Spanish documents

María-Isabel Limaylla-Lunarejo¹[0000–0002–9619–924X], Nelly
Condori-Fernandez^{2,3}[0000–0002–1044–3871], and Miguel R.
Luaces¹[0000–0003–0549–2000]

¹ Fac. Informática, Database Lab. Universidade da Coruña, CITIC, A Coruña, Spain
{maria.limaylla,miguel.luaces}@udc.es

² CITIUS, Universidad de Santiago de Compostela, Spain
n.condori.fernandez@usc.es

³ Vrije Universiteit Amsterdam, The Netherlands
n.condori-fernandez@vu.nl

Abstract. *Context and motivation:* Machine Learning (ML) algorithms and Natural Language Processing (NLP) techniques have effectively supported the automatic software requirements classification. The emergence of pre-trained language models, like BERT, provides promising results in several downstream NLP tasks, such as text classification. *Question/problem:* Most ML/DL approaches on requirements classification show a lack of analysis for requirements written in the Spanish language. Moreover, there has not been much research on pre-trained language models, like fastText and BETO (BERT for the Spanish language), neither in the validation of the generalization of the models. *Principal ideas/results:* We aim to investigate the classification performance and generalization of fastText and BETO classifiers in comparison with other ML/DL algorithms. The findings show that Shallow ML algorithms outperformed fastText and BETO when training and testing in the same dataset, but BETO outperformed other classifiers on prediction performance in a dataset with different origins. *Contribution:* Our evaluation provides a quantitative analysis of the classification performance of fastText and BETO in comparison with ML/DL algorithms, the external validity of trained models on another Spanish dataset, and the translation of the PROMISE NFR dataset in Spanish.

Keywords: Spanish requirements · automatic classification requirements · fastText · BETO

1 Introduction

Requirements specifications are significant activities that contribute to the success of a software project [16]. Manual classification of requirements is a complex and time-consuming task due to a large number of requirements, the unavailability of experts, or the lack of specification documents [3]. This complexity can

This is the author’s preprint of the following published work:

Limaylla-Lunarejo, M. I., Condori-Fernandez, N., & Luaces, M. R. (2023, April). Requirements classification using fast-text and beto in spanish documents. In International Working Conference on Requirements Engineering: Foundation for Software Quality (pp. 159-176). Cham: Springer Nature Switzerland.

be reduced using automatic text classification techniques, whose application has increased rapidly in recent years due to the emergence of ML algorithms [19] and NLP techniques [46]. However, it is the recent expansion of the Deep Neural Networks and the introduction of the Transfer Learning concept that has allowed the creation of pre-trained models, i.e. models with a pre-training phase to capture knowledge from several tasks that can be used later for target task with limited data [13]. Pre-trained language models, pre-trained models for NLP, have become beneficial for downstream NLP tasks, e.g. question answering, sentiment analysis, and summarization [31].

Several studies have used and analyzed various ML techniques for classifying software requirements. For instance, Abad et al. [1] proposed a pre-processing approach for improving the performance of some existing ML algorithms. Kurtanovic and Maalej [22] investigated how accurately they can automatically classify requirements employing lexical and syntactical features, and how to handle the imbalanced classes employing sampling strategies. fastText and BERT, pre-trained language models, have been used also in requirements classification [15, 23, 39], with optimistic results. Despite this interest in the use of ML techniques for classifying requirements, there is still a lack of analysis for the requirements written in Spanish. Spanish is currently the second native language of the world by the number of speakers, the second language by published scientific documents, and the third language by Internet usage after English and Chinese [17]. In the literature, we have found scarce research exploiting requirements written in Spanish. For example, Apaza et al., [4] propose a hybrid model for requirements elicitation, and De Arriba et al., [6] created a Spanish Twitter dataset for applying sentiment analysis). Moreover, standards like ISO/IEC 25010 are also available in Spanish. These initiatives suggest the increasing importance of the Spanish language in RE.

In this paper, we investigate the performance and generalization of two pre-trained language models, fastText and BETO, in comparison with ML and Deep Learning (DL) algorithms for classifying Spanish requirements. We also analyze the impact of the use of fastText and BETO embeddings in combination with the Convolutional Neural Networks (CNN) algorithm. Two Spanish datasets were used in the experiments: first, the PROMISE NFR dataset [38] translated into Spanish, and second, a Spanish requirements dataset presented in [27]. We used this last dataset to confirm the external validity of the models trained on translated PROMISE NFR. The main contributions of this paper are:

- A quantitative analysis of performance metrics to compare the fastText and BETO classifiers with ML/DL algorithms for requirements classification.
- Analyzing the external validity of trained models with a translated version of PROMISE NFR on another Spanish dataset.
- The translation of the PROMISE NFR dataset in Spanish.
- The use of fastText and BETO embeddings for the CNN algorithm.

The paper has been organized in the following way. Section 2 presents the related works on automatic requirements classification. Section 3 presents the

research questions and the corresponding methodology used in this research. Section 4 reports the experiments carried out and their results. Section 5 discusses the results from the previous section. The most relevant threats to validity are presented in Section 6. Finally, we conclude the paper in Section 7.

2 Related Work

Previous studies have described the use of ML algorithms to classify requirements. Many ML algorithms have been used for the classification of functional and non-functional requirements (FR/NFR) [12, 15, 22]. Table 1 shows the result of a review that we have performed of several studies related to FR/NFR classification. We found that some studies reported the use of the Shallow ML algorithms⁴, like Naive Bayes (NB), Decision Tree (DT), and Logistic Regression (LR); and the use of DL algorithms. Dias and Cordeiro (2020) [12] present a comparison to determine the best combination of some Shallow ML algorithms with text vectorization techniques like Bag of Words (BoW), Chi-Squared (χ^2), and Term Frequency and Inverse Document Frequency (TF-IDF), to classify requirements. For FR/NFR classification, the combination of TF-IDF and LR gave the best results. Support Vector Machine (SVM) gets an f1-score of 0.93 and 0.92 for FR/NFR classification in [22]. A decision tree algorithm was used for FR/NFR classification in [1], with a processed dataset (i.e. using Part of Speech and Regular Expressions for identifying features instead of using of text vectorization). A comparison between the Random Forest algorithm and the gradient boosting algorithm was performed in [24],

Algorithms based on Neural-networks also have been used recently in the Requirements Engineering (RE) process. Some of them are Recurrent Neural Networks (RNN), CNN, and pre-trained models, e.g. Transformers. CNN achieves an f1-score of 0.77 [30] for FR/NFR classification. Hey et al. (2020) [15] introduce NoRBERT, a language model based on Transformers, which fine-tunes BERT (a pre-trained model used for transfer learning). NoRBERT achieves an f1-score of 0.90 and 0.93 for FR/NFR classification. Tiun et al.(2020) investigate the performance using a complex neural classifier, finding an f1-score value of 0.928 using fastText [39]. The DBGAT model, a combination of BERT and graph attention network, was presented in [23]. In comparison with several algorithms, DBGAT achieved f1-scores of up to 0.91. A BiGru model obtained an f1-score of 0.94 in [2]. Finally, an ensemble of some DL algorithms was presented in [33].

In this review, we found that there is a lack of automatic classification and labeled datasets in other languages different from English. Even though some studies have presented methods to generate classification models trained in English for use in a target language, (e.g. Cross-Language Text Classification [45]), some studies show that models trained and tested in native language outperform better in text classifications [5, 44]. However, no relevant research on the automatic F/NF classification of requirements written in Spanish has been found.

⁴ defined by [18]: The *Shallow ML* contains simple artificial neural networks and other ML algorithms such as Support Vector Machine, Logistic Regression

Table 1: Summary of studies related to FR/NFR Classification

Ref.	Algorithms	D.	NLP techniques	Best results
[12]	SVM, KNN, MNB, LR	B	BoW/TF-IDF (χ^2)	LR with TF-IDF achieve an f1-score of 0.91.
[1]	C4.5 DT (J48 in Weka)	A	POS and entity tagging	Processed dataset: average precision of 0.95 and recall of 0.94.
[22]	SVM	A	Use of lexical features	SVM achieve an f1-score of 0.93 (FR) and 0.92 (NFR).
[32]	SVM, KNN	B	BoW	SVM achieve an f1-score of 0.90 and KNN an f1-score of 0.82.
[24]	Random Forest, Gradient Boosting	C	-	RF achieves an accuracy of 0.826
[30]	CNN	A	Word Embedding	CNN achieve an average f1-score of 0.77, precision 0.81 and recall 0.785 for all classes (12).
[15]	NoRBERT (Transfer Learning)	A	BERT (Transformers)	BERT achieve an f1-score of 0.90 (FR) and 0.93 (NFR).
[33]	Ensemble DL (LSTM, BiLSTM, GRU, and CNN)	A	Word Embedding	CNN achieves an f1-score of 0.93. The ensemble model achieves an f1-score of 0.96 per class as a weight ensemble, and 0.94 as mean ensemble.
[39]	NB, LG, SVM, CNN, fastText	A	Doc2Vec and Word2Vec	fastText achieves an f1-score of 0.928.
[23]	BERT and GAT	A,C	-	DBGAT achieve an f1-score of 0.92 (FR) and 0.96 (NFR)
[2]	BiGRU	A	Word Embedding	BiGRU with word model achieve an f1-score of 0.94

Legend: Ref: Reference, D: Datasets, A: PROMISE, B: PROMISE_exp [26], C: Others

Another finding is that considering pre-trained models are still new, their use is increasing in RE due to their promising results. Moreover, we found that most of the studies use the PROMISE NFR dataset, or a newer version, for training and testing, which does not allow us to determine the level of generalization that these approaches have.

3 Research Design

This section explains the research design used in this work. The goal of this paper is to compare the classification performance and generalization of fastText and BETO models with conventional algorithms using two Spanish datasets.

3.1 Research Questions and Metrics

- **RQ1:** How do the fastText and BETO classifiers with other ML/DL algorithms compare in performance using the PROMISE NFR dataset translated into Spanish?
- **RQ2:** How does the use of fastText and BETO embeddings in combination with the CNN algorithm affect the classification results?

- **RQ3:** To which extent the performance of our classification models trained with a Spanish version of PROMISE NFR differ from those trained with the original PROMISE NFR dataset?
- **RQ4:** Is the classification performance of the models trained using the translated PROMISE NFR preserved when tested on another dataset?

From our research questions (RQs), we identified the *classification performance* as our dependent variable and the requirements in the datasets as the independent variable. The most commonly used performance metrics for classification problems are accuracy (A), precision (P), recall (R), and f1-score (F_1).

3.2 Datasets

Datasets are an essential component of any ML experiment. In our research, we used two datasets: the PROMISE NFR dataset translated into Spanish, and a Spanish requirements dataset (called the second dataset).

PROMISE [38] is a repository used in most requirements classification research. It has 84 datasets, and the one used for requirement classification is NFR, with 625 requirement sentences, 255 requirements identified as functional and 370 as non-functional. For this research, we downloaded the *PROMISE NFR dataset* from [8] and translated it into Spanish. We decided to perform a translation of this dataset for two main reasons. First, this dataset has been used as a benchmark by many classification approaches. And second, even if there is a certain imbalance of sub-classes at the level of NFR, the classes are better balanced only considering a binary classification (FR/NFR). The translation of the PROMISE NFR dataset into Spanish was performed with the help of the Google Translate tool⁵, a state of art machine translation software, through the Python library deep-translator⁶. Google Translate was chosen as a translation tool because it is free, one of the most popular, and is more accurate when both languages are similar [43]. Then the translation was reviewed, line by line, by the first author, performing some adjustments. In this translation process, the appearance of the English language in some requirements was changed to the Spanish language. However, the proper names and acronyms were preserved. For example, “MDI form” was translated as “Formulario MDI”, and “Dr Susan Poslusny” as “La Dra. Susan Poslusny”. Punctuation was also added to some requirements where it was missing. Finally, the labels were preserved, like some names and versions of some browsers, identified as old (e.g. “Internet Explorer 5”). Of the 625 records, one was removed due to duplication, resulting in 254 functional requirements and 370 non-functional. The translated dataset and the code used in this research are published in Zenodo⁷.

The *second dataset* is the one presented in [27], and consists of 389 requirements, collected from final degree projects from the University of A Coruña,

⁵ <https://translate.google.es/>

⁶ https://github.com/prataffel/deep_translator

⁷ <https://doi.org/10.5281/zenodo.7311148>

and has 300 requirements labeled as functional and 89 requirements labeled as non-functional.

3.3 fastText and BETO models

The pre-trained language models have presented a promising performance in the Text Classification domain. The BERT model [11], provided by Google, is a pre-trained model and one of the state-of-art NLP tasks. It has been previously used for requirements classification (NoRBERT [15], [36]) with a good classification performance and generalization. We selected BETO [7], a BERT-based model, to take advantage that it was pre-trained on a big Spanish corpus (i.e., Wikipedia, Wikinews, ParaCrawl, EUBookshop, and OpenSubtitles) and in several NLP tasks. Moreover, BETO is one of the oldest monolingual models and therefore most used. BETO has been utilized in several Spanish text classifications (e.g., in the classification of radiological reports [41] and for detecting anorexia in social media [29]), in sentiment analysis [6,10] as a support for several RE tasks such as requirements elicitation or prioritization, and recently for Spanish requirements classification [27]. Two models (cased and uncased) of BETO are available. Since the requirements to be analyzed are written in formal language, case model⁸ was used for this analysis.

fastText [20] is an open-source library developed by Facebook Research for text classification and word embedding. fastText has been successfully used in various text classification tasks. Recently, it has been also applied in FR/NFR classification [39]. Umer et al. [42] reported good results with the combination of fastText and CNN on the automatic text classification process. fastText has the advantage of handling out-of-vocabulary words through the use of n-gram characters. It also allows the training of a supervised classifier with labeled data and provides automatic hyperparameter optimization. Besides that, fastText provides pre-trained word embedding libraries for 157 languages, including Spanish. The pre-trained model for Spanish used in this paper is cc.es.300.bin⁹, trained using CBOW with dimension 300 and character n-grams of length 5.

3.4 Research Method

In order to answer our research questions, two different experiments were performed. In the following, we explain the two experiments conducted:

Experiment 1. We used a similar procedure to the one proposed by Dalal and Zaveri (2011) [9]. The procedure consists of four steps: a) document pre-processing, b) feature extraction, c) ML selection, and d) training and testing the classifier. As a first step, a *pre-processing* of the requirements was performed. For text management, some Natural Language Processing (NLP) techniques were used. A tokenization task, a stopword removal, and a stemming task were carried out for each requirement, only for the Shallow ML algorithms. The NLTK

⁸ <https://huggingface.co/dccuchile/bert-base-spanish-wwm-cased>

⁹ <https://fasttext.cc/docs/en/crawl-vectors.html>

library¹⁰ was used for these three tasks. The second step is *feature extraction*, which was used to process the text on quantities and frequencies. Two groups of techniques were implemented: frequency-based and embedding-based. The frequency-based one includes the BoW and TF-IDF techniques. The scikit-learn library was used for text vectorization (i.e., the CountVectorizer tool for BoW and TfidfVectorizer for TF-IDF). Both techniques were tested with different values of n-grams: Unigram, Bigram, and Trigram, for every Shallow ML algorithm. The embedding-based one includes the fastText and BETO embeddings, transforming words into their corresponding word embeddings, using with the CNN algorithm. Regarding the third step, we *selected four Shallow ML algorithms*: Naive Bayes (NB), Gaussian Naive Bayes (GNB), Logistic Regression (LR), Support Vector Machine (SVM), and *three DL algorithms*: CNN, BETO, and fastText. The CNN architecture of our model is based on [21], consisting of a Word embedding Layer as an input layer, a One-dimensional Convolutional Layer, a Dropout Layer, a Polling Layer, a Dense Layer with a ReLU action function, and a Dense Layer with a sigmoid activation function as an output layer. The Transformer library¹¹ was used to obtain the BETO Tokenizer and the BETO pre-trained Model. We used (i) an architecture of four layers: an Input Layer, a BERT Encoder Layer, a Dropout Layer, and a Dense Layer for classification; and (ii) a set of hyperparameters. The library fastText¹² allows the training of a supervised classifier and provides an automatic hyperparameter optimization.

Finally, as a last step, *the classifiers were trained and tested*. This experiment was performed with the translated PROMISE NFR dataset, using only the binary classification (FR/NFR). We used the default hyperparameters for the Shallow ML algorithms, but use a set of several combinations for the DL algorithms (epochs, dropout, learning rate). Google Colaboratory¹³, a platform for programming and executing Python and with free access for GPUs, was used to train BETO, due to the necessity of major capacity. A GPU with 12.68GB of RAM was used. fastText was also trained in Colaboratory, even not needing the GPU, but for the facility in installation. The other algorithms were performed on a laptop computer with Intel(R) Core(TM) i5-1135G7 with 2.40GHz and 8GB of RAM. We saved the models (i.e. the outputs of the ML algorithms run on datasets) for later use. We calculated an average accuracy, precision, recall, and f1-score for each class (FR and NFR) using 10-fold cross-validation to evaluate the models' performance.

Experiment 2. In the second experiment, the models with the best f1-score values for each algorithm on experiment 1 were selected and used for testing on the second dataset. This provided a new set of metric values to evaluate whether these models can be generalized. To ensure the performance comparison among ML/DL models has statistical significance, we conducted a multiple hypotheses

¹⁰ <https://www.nltk.org/>

¹¹ <https://github.com/huggingface/transformers>

¹² <https://fasttext.cc/>

¹³ <https://colab.research.google.com/>

testing [34], performing first a Cochran’s Q test (omnibus test) and later a McNemar test statistics (post hoc tests). We considered a significance level (α) of 0.05 for Cochran’s Q test and applied a Bonferroni’s correction for the post hoc test. We used the library MLxtend that implements both tests in Python.

4 Experiments and Results

4.1 FR/NFR Classification using PROMISE NFR dataset (RQ1/RQ2)

Our analysis focuses on a comparison of fastText and BETO classifiers with a set of Shallow ML algorithms (i.e. NB, GNB, LR, and SVM) and one DL algorithm (CNN) to validate the classification performance in functional (F) and non-functional (NF) requirements. Regarding the Shallow ML algorithms in combination with the two text vectorization techniques (BoW, TF-IDF) and n-gram, 24 metrics set were obtained. All classification performance metrics are published in Zenodo¹⁴ and the highest values for each algorithm are presented in Table 2. Table 3 shows the same results for the CNN, fastText, and BETO classifiers, and CNN in combination with fastText and BETO embedding.

When analyzing the results we found that NB with BoW, LR with BoW, and SVM with TF-IDF combinations gave the highest f1-score for the non-functional classification (0.93), and the combination of NB with BoW achieved a better f1-score value for functional classification (0.9). Regarding the use of n-gram with the frequency-based text vectorization, almost all models obtained the best f1-score values using Bigram over Unigram and Trigram, except the NB algorithm that improves using Trigram. The use of n-gram with the NB algorithm has reported an increase in the classification accuracy for text classification in other domains [40]. The use of fastText embeddings with the CNN algorithms outperforms the other DL algorithms with an f1-score of 0.87 for functional classifications and 0.92 for non-functional classification, followed by the fastText classifier with a slight difference. The BETO classifier and the CNN with BETO were outperformed by the fastText classifier and the Shallow ML algorithms. In terms of time processing in training, the Shallow ML algorithms took less than 2 seconds for each one, the lowest time for all algorithms, but they need a longer pre-processing step. The CNN and the BETO algorithms took approx. 2 and 35 minutes respectively, but considering the combination of several hyperparameter values that were tested, the time was greater. The time processing of fastText to perform the automatic hyperparameters optimization was approx. 7 minutes.

Table 4 shows the hyperparameters that provided the best results in performance for the DL algorithms, after performing several tests with various combinations. CNN algorithm reduces its number of epochs by combining with fastText embedding but increases with BETO embedding. These values suggest that classification models based on Shallow ML algorithms have a good capacity for

¹⁴ <https://doi.org/10.5281/zenodo.7602116>

Table 2: Results of Shallow ML algorithms in combination with text vectorization techniques for FR/NFR classification using translated PROMISE NFR dataset (RQ1)

	10-fold													
	NB BoW		GNB BoW		GNB TFIDF		LR BoW		LG TFIDF		SVM BoW		SVM TFIDF	
	Trigram		Bigram		Bigram		Bigram		Bigram		Bigram		Bigram	
	F	NF	F	NF	F	NF	F	NF	F	NF	F	NF	F	NF
A	0.92		0.90		0.89		0.91		0.88		0.89		0.91	
P	0.90	0.93	0.87	0.93	0.86	0.92	0.90	0.92	0.94	0.86	0.86	0.90	0.91	0.91
R	0.91	0.92	0.91	0.90	0.89	0.90	0.88	0.94	0.77	0.97	0.86	0.91	0.87	0.94
F_1	0.90	0.93	0.88	0.91	0.87	0.91	0.89	0.93	0.84	0.91	0.85	0.91	0.89	0.93

Table 3: Results of DL algorithms for FR/NFR classification using translated PROMISE NFR dataset (RQ1/RQ2)

10-fold										
	CNN+fastText		CNN+BETO		CNN		fastText		BETO	
	F	NF	F	NF	F	NF	F	NF	F	NF
A	0.90		0.83		0.88		0.9		0.86	
P	0.90	0.91	0.81	0.85	0.88	0.88	0.89	0.90	0.84	0.89
R	0.86	0.93	0.78	0.87	0.82	0.92	0.85	0.93	0.80	0.88
F_1	0.87	0.92	0.79	0.86	0.85	0.90	0.87	0.91	0.81	0.89

classifying Spanish requirements (especially the non-functional label). However, it is also necessary to validate whether these models can be generalized.

4.2 Classification performance comparison between models trained in translated PROMISE and original PROMISE (RQ3)

To answer RQ3, we compared the results of our first experiment with other empirical studies that used PROMISE NFR in its original language (English). Seven studies from the list shown in Table 1 were chosen because they used the same algorithms and a similar dataset, and most of them reported results of well-known performance metrics. As our comparison was carried out in terms of the macro f1-score metric, we contacted the authors of two studies [12, 32] to clarify how the f1-score metric presented in their works had been calculated. Other studies reported the f1-score metric, and we calculated and obtained the corresponding values from other performance measures for those that did not report it. Table 5 presents the macro f1-score values and the difference between the results of the studies compared with our results. The column “Own” means the results of our experiment.

We found slight differences between our results and other studies concerning Shallow ML algorithms. Regarding the DL algorithms, the DL ensemble model proposed by [33] obtained an f1-score value of 0.94, a higher value for our CNN models (CNN and CNN+fastText). We compared the architecture of the CNN model with ours, finding a main difference in the use of two dropout layers, one before the convolutional layer and the other before the last dense layer; while in our model only one dropout layer after the convolutional layer. Finally,

Table 4: Best Hyperparameters for DL algorithms

	Dropout Rate	Feature Maps	Optimizer	Learning Rate	Epochs
CNN+fastText	0.75	100	Adam	0.01	10
CNN+BETO	0.75	100	Adam	0.001	300
CNN	0.75	100	Adam	0.001	20
BETO	0.2	-	Adam	1e-05	15
fastText	-	-	-	0.101	100

Table 5: Metrics Comparison between previous studies

Reference	Dataset	Algorithm	Macro F_1	Own	Difference	Own**	Dif**
[12]	B	NB	0.91	0.92	-0.01		
[12]	B	LR	0.91	0.91	0		
[12]	B	SVM	0.91	0.91	0		
[22]	A	SVM	0.93	0.91	0.02		
[32]	B	SVM	0.90	0.91	-0.01		
[30]	A	CNN	0.77	0.89	-0.12	0.90	-0.13
[33]	A	CNN	0.93	0.89	0.06	0.90	0.03
[33]*	A	CNN	0.94	0.89	0.07	0.90	0.04
[39]	A	fastText	0.92	0.89	0.03		
[15]	A	BERT	0.92	0.85	0.07		

Legend: A: PROMISE (English), B: PROMISE.exp (English), *ensemble model, **CNN+fastText

fastText [39] and NoRBERT [15] (developed with some custom code) get better results for FR/NFR classification than our models.

4.3 Testing trained models on the second dataset (RQ4)

Models Performance. We tested the trained models derived from the first experiment on the second dataset. The results are shown in Table 6 for Shallow ML algorithms and Table 7 for DL algorithms. Contrary to the results obtained in the previous experiment, the BETO model gave the best results, with an f1-score of 0.84 for functional and 0.65 for non-functional classification, followed by the CNN with fastText model, with an f1-score of 0.77 and 0.51. In general, we found that the values for accuracy and the f1-score were lower than those obtained in experiment 1. It can be seen from the data in both Tables that the precision values for the functional class are above 0.85 in most of the models, but with a value below 0.51 for the non-functional class. This result indicates a problem in the prediction of NFR since less than 50% of the requirements that the models predict as non-functional are really non-functional. An analysis of several cases containing a false value of the non-functional class was performed to understand and identify the possible causes of these misclassifications.

The use of some specific technical words in FR could lead to confusion for the models. The word “servidor” (“server” in English) is a word normally used in NFR, such as server availability, server failure, etc. In the PROMISE NFR dataset, 15 requirements mention this word and they are all non-functional. But in the second dataset, one of the projects includes some requirements (considered functional by the author) that contain also the word “server” with other technical words (API, PC), which are incorrectly predicted as non-functional. Another

Table 6: Results of testing the Shallow ML algorithms of experiment 1 on the second dataset (RQ4)

	NB + BoW Trigram Model 1		GNB+BoW Bigram Model 2		GNB+ TF-IDF Bigram Model 3		LR+BoW Bigram Model 4		LR+ TF-IDF Bigram Model 5		SVM+BoW Bigram Model 6		SVM+ TF-IDF Bigram Model 7	
	F	NF	F	NF	F	NF	F	NF	F	NF	F	NF	F	NF
A	0.64		0.65		0.58		0.66		0.46		0.64		0.64	
P	0.93	0.37	0.85	0.35	0.83	0.30	0.92	0.39	0.98	0.30	0.89	0.36	0.93	0.38
R	0.58	0.85	0.66	0.61	0.57	0.62	0.61	0.83	0.31	0.98	0.60	0.75	0.58	0.85
F_1	0.71	0.52	0.74	0.44	0.68	0.40	0.73	0.53	0.47	0.45	0.72	0.49	0.71	0.52

Table 7: Results of testing the DL algorithms of experiment 1 on the second dataset (RQ4)

	CNN Model 8		BETO Model 9		fastText Model 10		CNN+fastText Model 11		CNN+BETO Model 12	
	F	NF	F	NF	F	NF	F	NF	F	NF
A	0.61		0.78		0.61		0.68		0.60	
P	0.76	0.20	0.97	0.51	0.96	0.36	0.89	0.40	0.93	0.35
R	0.72	0.24	0.74	0.91	0.52	0.92	0.67	0.73	0.53	0.87
F_1	0.74	0.22	0.84	0.65	0.68	0.52	0.77	0.51	0.67	0.50

example is the word “autenticar” (“authenticate” in English). Ten requirements in the second dataset contain that word (or similar) and are labeled as functional, but the models classify them as non-functional. Other examples are “base de datos” and “archivo” (“database” and “file” in English, respectively). Another cause could be the lack of detail in the second dataset for NFR, in comparison with the PROMISE NFR dataset. In a previous analysis of this dataset [27], some ambiguity in the requirements specification was identified as another cause for misclassifications of the NFRs.

Models Comparison. We performed Cochran’s Q and McNemar’s test statistics to ensure the performance metrics comparison has statistical significance, testing the null hypothesis (all the model performance are similar), and the alternative hypothesis (all the model performance has significance difference). We combined some models from Table 6 and Table 7 to generate interesting groups for testing. The Cochran’s Q test statistics for model groups are shown in Table 8. These results show that there is a statistical significance in the difference between all the models’ performance, between the performance of the models based on Shallow ML algorithms, and between the performance of the models based on DL algorithms.

We performed some pair-wise McNemar’s tests on the models based on DL algorithms, shown in Table 9. We calculated a significance level of 0.001 using the Bonferroni alpha adjustment: considering $k = 10$, then the new value is $0.05/[k(k-1)/2] = 0.05/[10(9)/2] = 0.001$. The comparison of BETO with each of the other DL models demonstrated there are significant differences in the performance between BETO and these models. However, others pair-wise of CNN, fastText,

CNN with fastText, and CNN with BETO obtained a p-value greater than 0.001, showing that these models did not show significant differences among them.

Table 8: Cochran’s q test statistics

Models Groups	Q value	P value
All models shown in Table 6 (Models 1 - 12)	122.677	<0.05
Models based on Shallow ML algorithms (Models 1 - 7)	78.935	<0.05
Models based on DL algorithms (Models 8 - 12)	42.559	<0.05

Table 9: McNemar’s test statistics

Models Pairs	χ^2 value	P value
BETO and fastText	34.018	<0.001
BETO and CNN	23.141	<0.001
BETO and CNN+fastText	10.208	<0.001
BETO and CNN+BETO	33.767	<0.001
CNN and fastText	0.004	0.945
CNN and CNN+fastText	4.148	0.041
fastText and CNN+fastText	6.318	0.019
CNN and CNN+BETO	0.005	0.943

5 Discussion

Several studies have shown that Shallow ML algorithms, like SVM, present good performance when the training is carried out in small and homogeneous datasets. We found Shallow ML algorithms give better results than DL algorithms (including fastText and BETO) using a translated PROMISE NFR (RQ1). This finding supports the work of other studies [25,28] that have demonstrated a better performance of Shallow ML algorithms in small datasets. Although the datasets used in this research are considered Large-Scale Requirements Engineering (i.e., more than 100 and until 1000 requirements [35]), they are still considered small in comparison to other datasets used in text classification with ML techniques, like the IMDB reviews and Reuters dataset presented in [25]. Structured data is another factor that also influences these results since it is easier to train a Shallow ML model. In this research, the requirements specifications from the PROMISE NFR dataset are (semi-) structured.

The fastText classifier was not only the easier algorithm to implement, but also one of the most effective DL algorithms. Besides, it was the fastest algorithm among the DL algorithms. In comparison, BETO was the one with the longest time, due to its architecture and computation complexity, expressed as a limitation in [13,31]. In general, in experiment 1 the f1-score values are higher in

the non-functional over the functional classification, which confirms other similar studies that use PROMISE NFR in its original language [1, 15]. A possible explanation for these results is that the PROMISE NFR dataset has a detailed requirement description, especially the use of some cardinal numbers for the NFRs. Kurtanovic and Maalej [22] found that cardinal numbers are the best single informative feature.

We also validate the impact of the use of fastText and BETO embeddings (RQ2). The fastText embedding increments the performance when used as weights in the embedding layer in the CNN architecture and decreases the number of epochs in training. It is therefore likely that the use of fastText embedding is an influential factor in classification performance when using with CNN, according to other results in text classifications [37, 42].

Regarding the multi-language aspect in requirements classification, we found some differences between the English and Spanish languages in the classification process itself. The dataset search was the more challenging. We had to expand the search for research papers on both languages, consider other not well-known repositories, and a translation process had to be carried out, due a dataset of labeled requirements in the Spanish language was not found. Even though the techniques were the same, some resources for the Spanish language (e.g. the Snowball stemmer for Spanish from NLTK library) had to be investigated. Furthermore, a comparative review of our metrics' values was performed with previous studies that used the same algorithms and the same dataset in its original language (RQ3). The metrics values of the shallow ML algorithms are observably close, but with higher values for the DL algorithms in the English language. We suspect that different architectures of the DL models could be one of the factors explaining these gaps. Considering these differences, it can be assumed that the process used in this study can be used for other languages similar to Spanish (Romance languages).

A second dataset in Spanish with different origins was used to validate the generalization of the models obtained in experiment 1 (RQ4). All the models decrease the f1-score values for a functional classification except for the BETO classifier. According to these results, we can claim that the BETO classifier, even trained with a semi-structured and balanced dataset, can offer a better capability to classify unseen data (generalization). These results reflect those of Hao et al. (2019) [14] and Hey et al. (2020) [15], who also came up with the best-generalized models using BERT. Cochran's Q test and McNemar's test results confirm the higher performance of BETO over others models is statistically significant. Without considering BETO, the DL algorithms performed equally, but significantly less than BETO. It would be interesting to test our obtained BETO classifiers with other Spanish software repositories to confirm our results.

Finally, the results from experiment 2 also showed that the performance values are much lower in the non-functional classification, similar to the results obtained in [27]. This was mainly due to the low precision values. There may be several possible explanations for this result. One is that some technical words are used in the definitions of FR, misclassifying FR as NFR. Another factor

could be the lack of detail in the definition of the requirements, compared with PROMISE NFR. The use of n-grams could also make the shallow ML models more restrictive, even the DL models also present the same problem.

6 Threats to Validity

In this section, we present the threats to the validity of our research as well as the actions that were taken for their reduction or (partial) mitigation.

1. **Internal validity:** A common problem in automatic requirements classification is the lack of high-quality labeled datasets. Given that the PROMISE NFR dataset has been already used in previous works, we considered a translation to Spanish using Google Translate. However, this translation process can introduce some bias (e.g. gender bias), which can not be mitigated. For instance, in the English Language, the word “user” is applied for both gender, but in our translation, “usuario” represents only a neutral/masculine gender. Furthermore, it’s being considered in the future to use other translators such as DeepL¹⁵, and perform comparisons to ensure greater consistency in the translation.
2. **Construct validity:** The classification performance was evaluated by means of very well-known metrics (i.e., precision, recall and f1-score). However, working with ML models and small datasets like the PROMISE NFR dataset, there is a risk of overfitting the models. To avoid this risk, we used the 10-fold cross-validation technique and tested with different parameters. The code and dataset were shared in Zenodo, so other developers and/or researchers can emulate this work.
3. **External validity:** To increase the external validity of our results obtained from the first experiment, a second dataset created with different software projects to PROMISE was used to validate our obtained models. The requirements specifications from this dataset represent three project types (web, android, and broker) and several domains (e.g. Education, Media, Medical, Finance, etc.) to validate the generalizability of the trained models on PROMISE NFR. The experiment has shown that BETO (and to a lesser extent CNN+fastText) can achieve a good generalization performance on classifying requirements for different project types and domains. However, considering the diversity of the Spanish language in Spanish-speaking countries, more tests in these models need to be undertaken to ensure the generalization to other Spanish variants and other languages.

7 Conclusions

The purpose of the current study is to investigate the performance of the pre-trained models fastText and BETO in comparison with other traditional algorithms for automatic functional and non-functional requirements classification in

¹⁵ <https://www.deepl.com/translator>

Spanish datasets, through two experiments. Our findings from our first experiment, using the translated PROMISE NFR dataset, reveal that most of the Shallow ML outperform the DL algorithms. However, BETO performed significantly better on the second dataset (second experiment). This suggests that pre-trained language models can obtain better generalizability, i.e., can have a better capability for classifying new requirements, even when their specifications could not be very well detailed. Finally, even though the BETO classifier obtained the best performance in generalizability, the fastText classifier overperformed the others DL algorithms in time processing. Further research might continue to explore requirements classification in the Spanish language. As our results found that BETO offers the best generalized model, we plan to replicate the experiment but including other Spanish BERT-based algorithms. Another natural progression of this work is expanding our Spanish dataset with requirements from the industry. However, in order to identify potential companies interested in sharing their data (if exist), we plan to conduct some interviews in order to understand the actual practices regarding requirements documentation in Spanish-speaking organizations.

Acknowledgement

This research was partially funded 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).

References

1. Abad, Z.S.H., Karras, O., Ghazi, P., Glinz, M., Ruhe, G., Schneider, K.: What Works Better? A Study of Classifying Requirements. In: Proceedings - 2017 IEEE 25th International Requirements Engineering Conference, RE 2017 (2017). <https://doi.org/10.1109/RE.2017.36>
2. AlDhafer, O., Ahmad, I., Mahmood, S.: An end-to-end deep learning system for requirements classification using recurrent neural networks. *Information and Software Technology* **147**, 106877 (2022)
3. Alrumaih, H., Mirza, A., Alsalamah, H.: Toward automated software requirements classification. In: 2018 21st Saudi Computer Society National Computer Conference (NCC). pp. 1–6. IEEE (2018)
4. Apaza, R.D.G., Barrios, J.E.M., Becerra, D.A.I., Quispe, J.A.H.: Ers-tool: hybrid model for software requirements elicitation in spanish language. In: Proceedings of the International Conference on Geoinformatics and Data Analysis. pp. 27–30 (2018)
5. Plaza-del Arco, F.M., Molina-González, M.D., Urena-López, L.A., Martín-Valdivia, M.T.: Comparing pre-trained language models for spanish hate speech detection. *Expert Systems with Applications* **166**, 114120 (2021)
6. de Arriba, A., Oriol, M., Franch, X.: Applying transfer learning to sentiment analysis in social media. In: 2021 IEEE 29th International Requirements Engineering Conference Workshops (REW). pp. 342–348. IEEE (2021)

7. Cañete, J., Chaperon, G., Fuentes, R., Ho, J.H., Kang, H., Pérez, J.: Spanish pre-trained bert model and evaluation data. In: PML4DC at ICLR 2020 (2020)
8. Cleland-Huang, J., Mazrouee, S., Liguio, H., Port, D.: nfr [data set] (mar 2007). <https://doi.org/10.5281/zenodo.268542>, <https://doi.org/10.5281/zenodo.268542>
9. Dalal, M.K., Zaveri, M.A.: Automatic text classification: a technical review. *International Journal of Computer Applications* **28**(2), 37–40 (2011)
10. De Arriba, A., Oriol, M., Franch, X.: Merging datasets for emotion analysis. In: 2021 36th IEEE/ACM International Conference on Automated Software Engineering Workshops (ASEW). pp. 227–231. IEEE (2021)
11. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018)
12. Dias Canedo, E., Cordeiro Mendes, B.: Software requirements classification using machine learning algorithms. *Entropy* **22**(9), 1057 (2020)
13. Han, X., Zhang, Z., Ding, N., Gu, Y., Liu, X., Huo, Y., Qiu, J., Yao, Y., Zhang, A., Zhang, L., et al.: Pre-trained models: Past, present and future. *AI Open* **2**, 225–250 (2021)
14. Hao, Y., Dong, L., Wei, F., Xu, K.: Visualizing and understanding the effectiveness of bert. *arXiv preprint arXiv:1908.05620* (2019)
15. Hey, T., Keim, J., Koziol, A., Tichy, W.F.: Norbert: Transfer learning for requirements classification. In: 2020 IEEE 28th International Requirements Engineering Conference (RE). pp. 169–179. IEEE (2020)
16. Hussain, A., Mkpojiogu, E.O., Kamal, F.M.: The role of requirements in the success or failure of software projects. *International Review of Management and Marketing* **6**(7S), 306–311 (2016)
17. Instituto Cervantes: El español una lengua viva (2021), https://cvc.cervantes.es/lengua/espanol_lengua_viva/, [Online; accessed 30-November-2021]
18. Janiesch, C., Zschech, P., Heinrich, K.: Machine learning and deep learning. *Electronic Markets* **31**(3), 685–695 (2021)
19. Jindal, R., Malhotra, R., Jain, A.: Techniques for text classification: Literature review and current trends. *webology* **12**(2) (2015)
20. Joulin, A., Grave, E., Bojanowski, P., Mikolov, T.: Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759* (2016)
21. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1746–1751. ACL, Doha, Qatar (oct 2014). <https://doi.org/10.3115/v1/D14-1181>, <https://aclanthology.org/D14-1181>
22. Kurtanovic, Z., Maalej, W.: Automatically Classifying Functional and Non-functional Requirements Using Supervised Machine Learning. In: Proceedings - 2017 IEEE 25th International Requirements Engineering Conference, RE 2017 (2017). <https://doi.org/10.1109/RE.2017.82>
23. Li, G., Zheng, C., Li, M., Wang, H.: Automatic requirements classification based on graph attention network. *IEEE Access* **10**, 30080–30090 (2022)
24. Li, L.F., Jin-An, N.C., Kasirun, Z.M., Chua, Y.P.: An empirical comparison of machine learning algorithms for classification of software requirements. *International Journal of Advanced Computer Science and Applications* **10**(11) (2019)
25. Li, Q., Peng, H., Li, J., Xia, C., Yang, R., Sun, L., Yu, P.S., He, L.: A survey on text classification: From shallow to deep learning. *arXiv preprint arXiv:2008.00364* (2020)

26. Lima, M., Valle, V., Costa, E., Lira, F., Gadelha, B.: Software engineering repositories: expanding the promise database. In: Proceedings of the XXXIII Brazilian Symposium on Software Engineering. pp. 427–436 (2019)
27. Limaylla-Lunarejo, M.I., Condori-Fernandez, N., Luaces, M.R.: Towards an automatic requirements classification in a new spanish dataset. In: 2022 IEEE 30th International Requirements Engineering Conference (RE). pp. 270–271. IEEE (2022)
28. Liu, S.: Sentiment analysis of yelp reviews: a comparison of techniques and models. arXiv preprint arXiv:2004.13851 (2020)
29. López-Úbeda, P., Plaza-del Arco, F.M., Díaz-Galiano, M.C., Martín-Valdivia, M.T.: How successful is transfer learning for detecting anorexia on social media? *Applied Sciences* **11**(4), 1838 (2021)
30. Navarro-Almanza, R., Juarez-Ramirez, R., Licea, G.: Towards supporting software engineering using deep learning: A case of software requirements classification. In: 2017 5th International Conference in Software Engineering Research and Innovation (CONISOFT). pp. 116–120. IEEE (2017)
31. Qiu, X., Sun, T., Xu, Y., Shao, Y., Dai, N., Huang, X.: Pre-trained models for natural language processing: A survey. *Science China Technological Sciences* **63**(10), 1872–1897 (2020)
32. Quba, G.Y., Al Qaisi, H., Althunibat, A., AlZu'bi, S.: Software requirements classification using machine learning algorithm's. In: 2021 International Conference on Information Technology (ICIT). pp. 685–690 (2021). <https://doi.org/10.1109/ICIT52682.2021.9491688>
33. Rahimi, N., Eassa, F., Elrefaei, L.: One-and two-phase software requirement classification using ensemble deep learning. *Entropy* **23**(10), 1264 (2021)
34. Raschka, S.: Model evaluation, model selection, and algorithm selection in machine learning. arXiv preprint arXiv:1811.12808 (2018)
35. Regnell, B., Svensson, R.B., Wnuk, K.: Can we beat the complexity of very large-scale requirements engineering? In: International Working Conference on Requirements Engineering: Foundation for Software Quality. pp. 123–128. Springer (2008)
36. Sainani, A., Anish, P.R., Joshi, V., Ghaisas, S.: Extracting and classifying requirements from software engineering contracts. In: 2020 IEEE 28th international requirements engineering conference (RE). pp. 147–157. IEEE (2020)
37. Santos, I., Nedjah, N., de Macedo Mourelle, L.: Sentiment analysis using convolutional neural network with fasttext embeddings. In: 2017 IEEE Latin American conference on computational intelligence (LA-CCI). pp. 1–5. IEEE (2017)
38. Sayyad Shirabad, J., Menzies, T.: The PROMISE Repository of Software Engineering Databases. School of Information Technology and Engineering, University of Ottawa, Canada (2005), <http://promise.site.uottawa.ca/SERepository>
39. Tiun, S., Mokhtar, U., Bakar, S., Saad, S.: Classification of functional and non-functional requirement in software requirement using word2vec and fast text. In: journal of Physics: conference series. vol. 1529, p. 042077. IOP Publishing (2020)
40. Tripathy, A., Agrawal, A., Rath, S.K.: Classification of sentiment reviews using n-gram machine learning approach. *Expert Systems with Applications* **57**, 117–126 (2016)
41. Úbeda, P.L., Díaz-Galiano, M.C., López, L.A.U., Martín-Valdivia, M.T., Martín-Noguerol, T., Luna, A.: Transfer learning applied to text classification in spanish radiological reports. In: Proceedings of the LREC 2020 Workshop on Multilingual-BIO 2020. pp. 29–32 (2020)
42. Umer, M., Imtiaz, Z., Ahmad, M., Nappi, M., Medaglia, C., Choi, G.S., Mehmood, A.: Impact of convolutional neural network and fasttext embedding on text classification. *Multimedia Tools and Applications* pp. 1–17 (2022)

- 43. Vanjani, M., Aiken, M.: A comparison of free online machine language translators. *J. Manag. Sci. Bus. Intell* **5**, 26–31 (2020)
- 44. Virtanen, A., Kanerva, J., Ilo, R., Luoma, J., Luotolahti, J., Salakoski, T., Ginter, F., Pyysalo, S.: Multilingual is not enough: Bert for finnish. *arXiv preprint arXiv:1912.07076* (2019)
- 45. Xu, R., Yang, Y.: Cross-lingual distillation for text classification. *arXiv preprint arXiv:1705.02073* (2017)
- 46. Zhao, L., Alhoshan, W., Ferrari, A., Letsholo, K.J., Ajagbe, M.A., Chioasca, E.V., Batista-Navarro, R.T.: Natural language processing (nlp) for requirements engineering: A systematic mapping study. *arXiv preprint arXiv:2004.01099* (2020)