Journal of King Saud University – Computer and Information Sciences 35 (2023) 101665

HOSTED BY

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

FNReq-Net: A hybrid computational framework for functional and non-functional requirements classification

Summra Saleem [a,b,*], Muhammad Nabeel Asim [b,*], Ludger Van Elst [b], Andreas Dengel [a,b]

[a] Department of Computer Science, RPTU Kaiserslautern-Landau, Kaiserslautern 67663, Germany
[b] German Research Center for Artificial Intelligence GmbH, Kaiserslautern 67663, Germany

ARTICLE INFO

Article history:
Received 5 May 2023
Revised 18 July 2023
Accepted 19 July 2023
Available online 29 July 2023

Keywords:
Software development
Functional & non-functional requirements
Feature selection
Feature pruning
Attention mechanism
Hybrid predictor

ABSTRACT

Requirements classification is a key component of software development life cycle. It enhances our understanding about project requirements, which in turn enables us to effectively identify and mitigate risks that could lead to project failure. Existing requirements classification predictors do not utilize feature selection methods competence in their predictive pipelines and lack in performance. To empower the process of automatic requirements classification, contributions of this paper are manifold. Firstly, it explores the potential of 7 filter-based feature selection techniques and 11 traditional machine learning classifiers. Secondly, for the first time it investigates combined potential of traditional feature selection and 9 diverse types of deep learning predictors. Thirdly, it presents a hybrid computational predictor namely FNReq-Net that reaps combine benefits of traditional feature selection and a novel deep learning predictor based on attention mechanism. Over two public benchmark datasets, large-scale experimental results reveal feature selection not only improves predictive performance of traditional machine learning predictors, but it also improves performance of deep learning predictors. The proposed FNReq-Net predictor outperforms state-of-the-art functional and non-functional requirements classification predictors by 4% and 1% in terms of F1-score over Promise and Promise-exp datasets, respectively.

1. Introduction

An extended version of waterfall model known as v-model comprises of 2 different modules, namely development and testing (Petersen et al., 2009; Balaji and Murugaiyan, 2012). To make sure proper development of a software, v-model facilitates development and testing of software at different stages in a parallel fashion. Among all software development stages (Felderer and Travassos, 2020), requirements analysis is an indispensable stage, as it defines fundamental details of software from customer's perspective (Vogelsang and Borg, 2019). A pitfall at requirements analysis stage can propagate in all later development stages. In the last 4 decades, software development and requirement engineering (RE) related major research has been devoted for understanding how to identify, analyze and evaluate functional and non-functional requirements (Maruping and Matook, 2020; Hidellaarachchi et al., 2021). Primarily, requirements fall into 2 main categories: functional and non-functional. Functional requirements define specific features and functionalities that a software system must have. This class contains requirements related to user authentication, data entry, report generation, payment processing and search functionality. Non-functional requirements focus on the qualities and characteristics of a software system, such as performance, scalability, usability and security. This class contains requirements related to fast response times, ability to handle increased user load, intuitive user interface and robust security measures (Becker et al., 2019; Horkoff, 2019). Specifically, functional requirements focus on behavior, while non-functional requirements address qualities and characteristics beyond core functionality.

Requirements classification into functional and non-functional classes is essential to understand customers' expectations and to follow exact requirements. However, manual classification of requirements requires an expert person who has a profound understanding of functional and non-functional qualities of software at unit and system levels. Furthermore, requirements vary

* Corresponding authors.
E-mail addresses: summra.saleem@dfki.de (S. Saleem), muhammad_nabeel.asim@dfki.de (M.N. Asim).
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.jksuci.2023.101665
1319-1578/

Table 1
A comprehensive summary of existing research studies for requirements classification.

Authors	Dataset	Feature encoding technique	Predictor
Luo et al. (2022)	Promise, NFR-review, NFR-so	BERT	Prompt learning using BERT
Li et al. (2022)	Promise, Concordia	Node embedding	Graph attention network
Kaur and Kaur (2022)	Promise, Open source project	Glove	Self-attention based bi-directional RNN
Ivanov et al. (2022)	Pure	Fasttext, Elmo, BERT	SVM, BERT
Khayashi et al. (2022)	Pure	Glove, keras word embeddings	CNN, LSTM, BiLSTM, GRU, BiGRU
Ajagbe and Zhao (2022)	Promise, Pure, App review dataset, Google play store reviews	BERT based embeddings	Fine-tuned BERT
Kici et al. (2021a)	Doors, Promise, Pure	BERT embeddings	BERT, DistillBERT, Roberta, Al-BERT, XLNet
Althanoon and Younis (2021)	Promise	TFIDF	MNB, LR
Rahimi et al. (2021)	Promise	Random word embeddings	Ensemble learning using CNN, LSTM, BiLSTM, GRU
Kici et al. (2021b)	Doors, Promise	-	DistillBERT
Fávero and Casanova (2021)	Open source project	BERT based embeddings	Fine-tuned BERT
Tiun et al. (2020)	Promise	Word2vec, FastText	CNN
Dias Canedo and Cordeiro Mendes (2020)	Promise-exp	BoW, TFIDF	LR, SVM, KNN, MNB
Rahimi et al. (2020)	Self-collected	TFIDF	Ensemble of SVM, SVC, LR, NB, DT
Hey et al. (2020)	Promise	BERT based embeddings	Fine-tuned BERT
Rahman et al. (2019)	Promise	Word2vec	RNN, LSTM, GRU
Baker et al. (2019)	Promise	Random word embeddings	ANN, CNN
Haque et al. (2019)	Promise	BoW, TFIDF	NB, SVM, DT, KNN
Tóth and Vidács (2019)	Promise, Stack overflow dataset	TFIDF	BernoulliNB, DT ET, ETs, KNN, Label propagation, Label spread, LR, MLP, MNB, SVM

from software to software and raise the chance of error by incorrectly categorizing functional requirements into non-functional requirements and vice versa (Knauss et al., 2011). Apart from classification error, manual classification of requirements is an expensive and time-consuming task.

With an aim to automate the process of requirements classification, several rule-based predictors (Vlas and Robinson, 2011; Vlas and Robinson, 2012; Singh et al., 2016) have been proposed but these predictors require additional rules based on specifications of particular software (Vlas and Robinson, 2012; Jarzebowicz and Weichbroth, 2021). Following the success of machine learning approaches in diverse types of application areas including computer vision (CV) (Haghighat and Sharma, 2023), natural language processing (NLP) (Khurana et al., 2023; Saleem et al., 2022) and speech analysis, (Chaiani et al., 2022) researchers have utilized the power of machine learning methods for the development of generalized requirements classification predictors (Hey et al., 2020; Althanoon and Younis, 2021; Dias Canedo and Cordeiro Mendes, 2020; Kaur and Kaur, 2022). In the marathon of developing more robust and precise requirements classification predictors, according to the best of our knowledge in last 5 years, 19 predictors have been developed (Luo et al., 2022; Li et al., 2022; Kaur and Kaur, 2022; Ivanov et al., 2022; Khayashi et al., 2022; Ajagbe and Zhao, 2022; Kici et al., 2021a; Althanoon and Younis, 2021; Rahimi et al., 2021; Kici et al., 2021b; Fávero and Casanova, 2021; Tiun et al., 2020; Dias Canedo and Cordeiro Mendes, 2020; Rahimi et al., 2020; Hey et al., 2020; Rahman et al., 2019; Baker et al., 2019; Haque et al., 2019; Tóth and Vidács, 2019).

Table 1 provides a comprehensive overview of existing predictors in terms of representation learning and classification algorithms. while developing traditional machine learning based predictors researchers' focus was to improve classification performance by ensembling multiple predictors. Although, it is widely accepted that feature selection enhances the performance of machine learning classifiers. However, in requirement classification their potential remains unexplored. In natural language processing for text document classification (Parlak and Uysal, 2023; Jalal et al., 2022), sentiment analysis (Sharma and Jain, 2023; Kumar et al., 2022) and information retrieval (Abbasi et al., 2022), researchers have performed large-scale studies to benchmark the performance of traditional feature selection methods and machine learning classifiers. Similarly, in bio-informatics for genomics, proteomics and omics analysis tasks (Zanella et al., 2022; Lualdi and Fasano, 2019; Leclercq et al., 2019) researchers have explored the potential of feature selection strategies. Primarily, the aim of these studies has been to unlock the potential of feature selection algorithms for the development of more comprehensive real-world applications. In requirement classification predictive pipelines, incorporation of feature selection methods may enhance predictors performance.

On the other hand in existing deep learning based requirement classification predictors primary focus of researchers was to utilize the potential of CNN and RNN architectures. Usually, requirement classification datasets are small and traditional deep learning predictors may not produce performance similar to their performance on other text classification tasks where a large training data is available. Rather than focusing on CNN and RNN architectures that require large training data, lightweight predictors based on attention mechanisms may be more useful here. Primarily, attention architecture will help predictor in focusing on more discriminative features and may improve the performance of predictor. The prime focus of this research is to validate diverse types of hypotheses and develop a more robust and precise predictor for functional and non-functional requirements classification. Mainly, research of paper in hand revolves around 4 different hypotheses.

- Hypothesis 1: Do traditional feature selection methods improve machine learning classifier's performance?
  Validation: To analyze this hypothesis we explored the potential of 11 traditional machine learning classifiers along with 7 most widely used filter-based feature selection methods namely; Bi-Normal Separation (BNS) (Forman et al., 2003), Nor-

malized Difference Measure (NDM) (Rehman et al., 2017), Min–Max Ratio (MMR) (Rehman et al., 2018), Balanced Accuracy Measure (ACC2) (Forman et al., 2003), Poison Ratio (POISON) (Gao and Wang, 2009), Chi-Squared (CHISQ) (Resnik, 1999) and Gini-Index (GINI) (Park et al., 2010).

- Hypothesis 2: Which optimal combination of feature selection algorithm and classifier produces better performance?.
  Validation: We perform large-scale performance analysis over two public benchmark datasets and analyze the performance of feature selection and machine learning classifiers.
- Hypothesis 3: Dose external feature engineering improve predictive performance of deep learning classifiers?.
  Validation: We perform large-scale experimentation over 2 public benchmark datasets by using 9 deep learning based predictors with all input features and by selecting the most informative features.
- Hypothesis 4: Dose combined potential of traditional feature selection and attention architecture enhance requirement classification performance by focusing on most informative features?.
  Validation: We design a novel deep learning classifier by utilizing traditional feature selection method and attention-based architecture.

The remaining sections of manuscript are organized into 8 different sections where Section 2 summarizes existing research studies for requirement classification. Section 3 demonstrates details of proposed framework along with benchmark datasets and evaluation measures. Sections 4 and 5 provide comprehensive details about experiment setup and results, respectively. Sections 6 and 7 highlight research findings and limitations of proposed framework. Lastly, Section 8 presents the overall analysis and future directions.

2. Related work

This section briefly discusses diverse types of machine and deep learning based predictors that have been proposed for requirement classification. According to best of our knowledge, in last 4 years 4 machine learning predictors are developed for requirements classification (Althanoon and Younis, 2021; Dias Canedo and Cordeiro Mendes, 2020; Haque et al., 2019; Tóth and Vidács, 2019). These predictors made use of traditional machine learning classifiers (Multinomial Naive Bayes (MNB), Logistic Regression (LR), Support Vector Machine (SVM), Stochastic Gradient Descent (SGD), Bernouli Naive Bayes (BNB), Gaussian Naive Bayes (GNB), Decision Tree (DT) and K-Nearesrt Neighbor (KNN)) and TFIDF (term frequency-inverse document frequency) based feature representation approach. Apart from standalone classifiers based predictors, Rahimi et al. (Rahimi et al., 2020) developed a meta-predictor that reaped the benefits of 5 different machine learning classifiers (LR, SVC, SVM, DT and NB) and TFIDF based representation approach. While developing these predictors, prime focus of researchers was to explore the potential of different machine learning classifiers. It is widely accepted that traditional machine learning classifiers lack feature engineering and requires an external algorithm for removing irrelevant and redundant features. In a nutshell, these predictors lack feature selection strategy in their predictive pipelines.

Following the success of deep learning approaches in diverse types of NLP tasks, 4 different deep learning predictors have been proposed (Tiun et al., 2020; Rahimi et al., 2021; Khayashi et al., 2022; Kaur and Kaur, 2022). In two different predictors (Tiun et al., 2020; Kaur and Kaur, 2022) objective of researchers was to explore the potential of word embedding methods (word2vec, fasttext and glove) and deep learning architectures including convolutional neural network (CNN) and long-short term memory (LSTM). Objective of 2 predictors (Rahimi et al., 2021; Khayashi et al., 2022) was to reap combined benefits of different neural architectures and develop a meta-predictor. These meta-predictors are designed in multi-branching fashion, where each branch predicts class of requirements based on the unique architecture and finally all branches' probabilities are utilized to predict final class. Primarily, CNN and recurrent neural network (RNN) based deep learning predictors perform better with large data and usually requirements data are small. In diverse types of NLP tasks, researchers have proved that in deep learning predictors, utilization of attention architectures significantly improves their predictive performance. However, in existing requirements classification predictors, prime focus of researchers was to develop meta predictors and potential of attention architectures remains unexplored.

Researchers have explored the potential of transformer based language models (BERT, AL-BERT, Roberta, DistilBERT and XLNet) to generate comprehensive feature representations of requirements (Fávero and Casanova, 2021; Ajagbe and Zhao, 2022; Hey et al., 2020; Luo et al., 2022; Kici et al., 2021a). Apart from traditional transformer based language models, Li et al. (2022) predictor reaped the benefits of graph strategy and bert based language model for requirements classification. An in-depth performance analysis of pre-trained and fine-tuned models demonstrated that generalized feature representation models remained fail to effectively encode domain-specific terms of software engineering.

3. Materials and methods

The prime focus of this paper is to explore the potential of traditional feature selection approaches for the development of robust and precise machine and deep learning based end-to-end predictive pipelines for functional and non-functional requirements classification. The following subsections briefly summarize machine and deep learning based predictive pipelines.

3.1. Machine learning predictors

Fig. 1 graphically illustrates details of machine learning framework. The framework consists of 4 different modules namely; preprocessing, feature selection, feature representation and classification. Pre-processing module normalizes corpus text and removes punctuations and extra spaces. Furthermore, it performs feature pruning, where it takes all samples of corpus and removes words that occur in the corpus samples greater than selected upper threshold and words which occur less than selected lower threshold. Here, upper and lower thresholds are hyper-parameters that need to be set according to nature and complexity of task. Normally, upper threshold is set 85% percent and lower threshold is set at 25%. Specifically, it is considered that words that occur in more than 85% samples of the corpus are not discriminative and should be removed and similarly the words that occur in less than 25% samples of the corpus could be discriminative but they are rare words and may confuse classifiers so they should be removed.

Feature selection module removes irrelevant and redundant features (Tutsoy et al., 2020). Machine learning classifiers learn discriminative patterns among different classes and based on discriminative patterns they categorize unseen samples into pre-defined classes. When machine learning classifiers are fed with samples having irrelevant and redundant features, their performance deteriorates. On the other hand, even a simple classifier performs better when it is fed with the most informative and discriminative features.

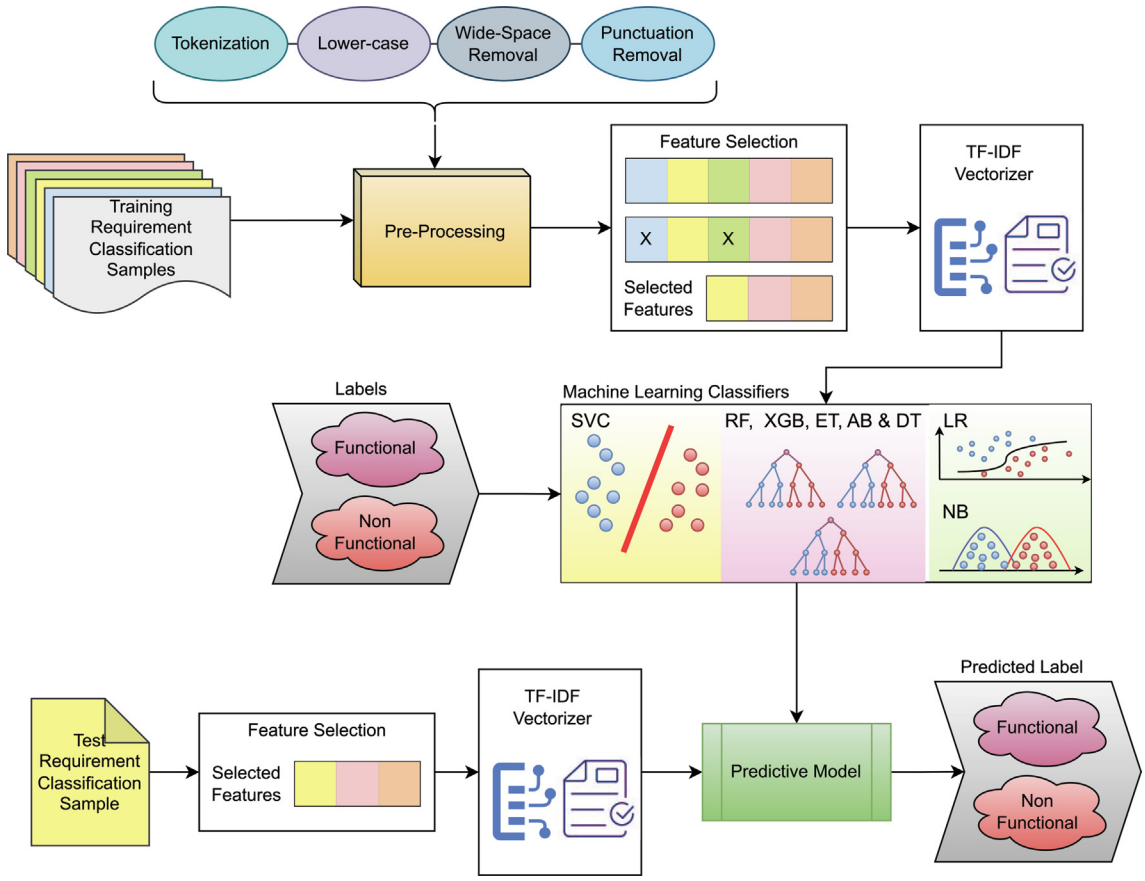

Fig. 1. Graphical illustration of traditional feature selection and machine learning classifiers based predictive pipeline for requirement classification. First of all data is split into train and test sets and per-processing is performed. After that in train set, requirement samples and their associated class labels are utilized to perform feature selection and further requirement samples with selected features are used to compute TFIDF representation that is used to train machine learning classifiers and trained classifier is saved and we named it as predictive model. Furthermore, train set based selected features are only retained in test set and test set samples TFIDF representation is computed and further passed to predictive model that provides class labels associated with the required samples.

With an aim to boost predictive performance of classifiers by feeding them with the most informative features, researchers have proposed diverse types of feature selection algorithms (Wang et al., 2017). Mainly, existing feature selection algorithms are categorized into 3 different classes including; filter (Jović et al., 2015), wrapper (El Aboudi and Benhlima, 2016) and embedded (Maldonado and López, 2018). Different types of feature selection methods that fall into wrapper and embedded categories use classifiers to iteratively elect a feature set. These methods are computationally expensive and can be utilized for tasks where corpus contains a small set of features and cannot be applied for classification tasks where corpus contains numerous features. In contrast, filter-based feature selection methods do not require any classifier for the selection of informative features and utilize metric computation for the selection of most informative features.

Requirement classification task contains thousands of features, so it is computationally not suitable to apply wrapper or embedded based feature selection methods. We enriched feature selection module with 7 different filter-based feature selection methods: ACC2 (Forman et al., 2003), BNS (Forman et al., 2003), CHISQ (Resnik, 1999), GINI (Park et al., 2010), MMR (Rehman et al., 2018), NDM (Rehman et al., 2017), POISON (Gao and Wang, 2009)). Comprehensive details of these methods can be found in cited articles. Feature representation module takes selected features based requirement samples and makes use of TFIDF feature representation method to transform textual samples into statistical vectors. The classification module takes TFIDF based statistical representation and class labels for training machine learning classifiers and final trained classifiers are fed with only statistical vectors and they predict class labels associated with statistical vectors. The classification module is enriched with 11 different traditional machine learning classifiers: AdaBoost (Margineantu and Dietterich, 1997), Decision Tree (Quinlan, 1996), Extra Tree (Geurts et al., 2006), Gaussian NB (Leung et al., 2007), Gradient Boosting (Friedman, 2021), KNeighbour (Aha, 1997), Logistic Regression (LaValley, 2008), Naive Bayes (Leung et al., 2007), Random Forest (Breiman, 2001), Support Vector Machine (Hearst et al., 1998), Extreme Gradient Boosting (Chen et al., 2015). Comprehensive details of these methods can be found in cited articles.

3.2. Adapted deep learning predictors

It is widely accepted that traditional machine learning classifiers performance relies on comprehensive feature selection (Shafiq et al., 2020) while deep learning predictors have inherent potential for automatically selecting informative features (Tang et al., 2019; Khamparia et al., 2021). However, recently it has been discovered that performance of deep learning predictors can be further improved by feeding them with most informative features selected through traditional feature selection strategies (Asim et al., 2019; Asim et al., 2021). A prime objective of this paper is to validate a unique hypothesis whether deep learning predictors are more competent in discriminating functional and non-functional requirements when they are fed with most informative

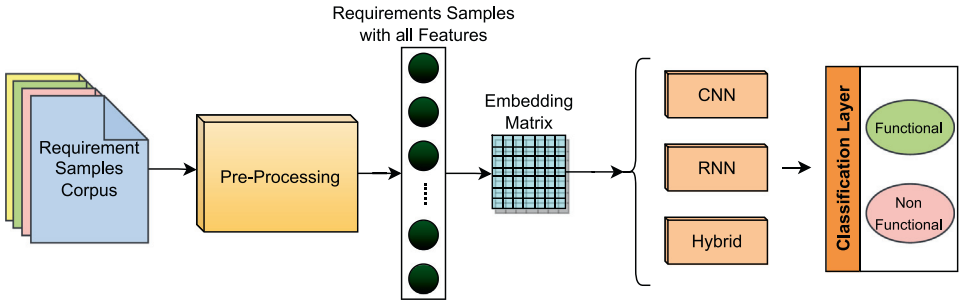

Fig. 2. Graphical illustration of deep learning-based predictors for requirements classification.

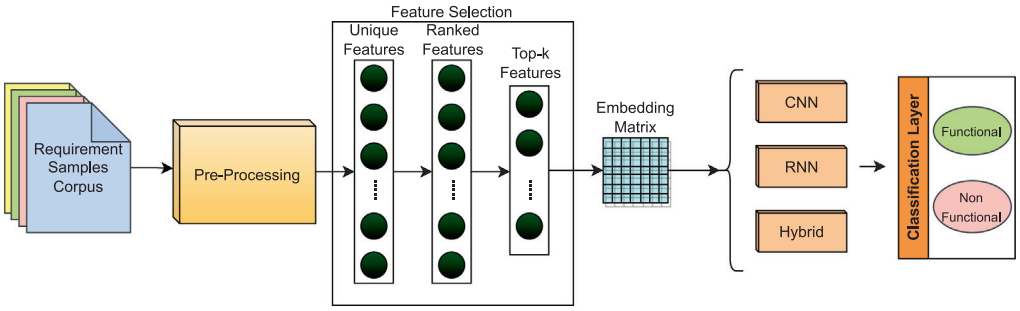

Fig. 3. Graphical illustration of traditional feature selection and deep learning predictors based predictive pipeline for requirements classification.

features as compared to conventional way in which they are fed with all input features. It can be seen in Table 1, only a few CNN, RNN and both CNN and RNN based hybrid predictors are designed for requirement classification. On the other hand, for textual document classification (Minaee et al., 2021) and sentiment analysis tasks (Onan, 2021) several deep learning predictors are proposed (Yogatama et al., 2017; Kalchbrenner et al., 2014; Chen, 2015). The paper in hand adapts 9 different deep learning predictors from textual document classification and sentiment analysis tasks, where aim is to analyze their performance for the task of requirement classification under two different experimental settings. Firstly by feeding models with requirement samples having all input features and secondly by feeding requirement samples having the most discriminative features. Figs. 2 and 3 illustrate generic requirements classification workflow of deep learning predictors in experimental settings 1 and 2, respectively.

Furthermore, comprehensive detail about adapted predictors and motivation behind their designed architecture is available in their original papers. As in this paper, their detailed explanation is beyond the scope so we have summarized details of predictors in Table 2. Interested readers can read further details about predictors in the manuscripts cited in Table 2. Further, to fully utilize the potential of these predictors, we performed experimentation by tweaking their number of layers and hyperparameters. The final architectural details of adopted predictors in terms of number of CNN, LSTM, pooling and dense layers are summarized in Table 2. Further, hyperparameters detail is provided in experimental setup section.

3.3. Proposed predictor

With an aim to more precisely distinguish requirements between functional and non-functional classes, we propose a hybrid predictor which reaps the benefits of traditional feature selection approach and deep learning-based attention mechanism for feature engineering. The complete working paradigm of the proposed predictor is graphically shown in Fig. 4.

After pre-processing input sentences, proposed predictor first generates a vocabulary of unique terms and features ranking module of proposed predictor ranks unique terms based on their discriminative power between both classes. After feature ranking, it selects the top k-ranked features from both classes and in each requirement sample generates their bi-grams, tri-grams and tetra-grams. Generated n-grams are passed to embedding layer, which randomly generates statistical vectors of n-grams. The proposed predictor generates dense comprehensive representation of features by taking average of n-gram based feature representation. The comprehensive feature representations are fed to self-attention layer that assigns higher weights to more informative features. The workflow of self-attention layer consists of multiple steps. First, attention layer applies linear operation on input sentence matrix x_{inp} and breaks it into three sub-matrices: query $q \epsilon R^{y \times z}$, key $k \epsilon R^{y \times z}$ and value $v \epsilon R^{y \times z}$, which is mathematically expressed in Eq. 1.

$$[q, k, v] = divide[x_{inp} \cdot w] \tag{1}$$

Here, $w \epsilon R^{z \times 3z}$ represents transformed matrix and *divide* refers to split function. Afterward, query and key matrices are multiplied to find similarity and output is known as attention filter. At first, weights of attention filters are randomly initialized, during iterative training these weights are updated to meaningful insight called attention score. Attention score is scaled by dividing with dimension of key metrics $\sqrt{z_k}$ and further normalized between range of 0 to 1 using softmax function. Finally, attention filter is multiplied by value metrics and fed to linear layer to compute attention score which is mathematically shown in Eq. 2.

Table 2
A comprehensive summary of adapted deep learning predictors.

Author	CNN Layers	LSTM Layers	No. of Channels	Pooling Layers	Dense layers
Kim (2014)	4	-	2	1-max pooling	2
Kalchbrenner et al. (2014)	2	-		2 k-max pooling	1
Yin and Schütze (2016)	4	-	2	1 max pooling	1
Zhang et al. (2016a)	6	-	3	-	1
Yogatama et al. (2017)	-	1	-	-	1
Palangi et al. (2016)	-	1	-	-	1
Chen et al. (2017)	5	1	-	1 max pooling	1
Zhou et al. (2015)	5	1	-	1 max pooling	1
Wang et al. (2016)	3	1	-	1 max pooling	2

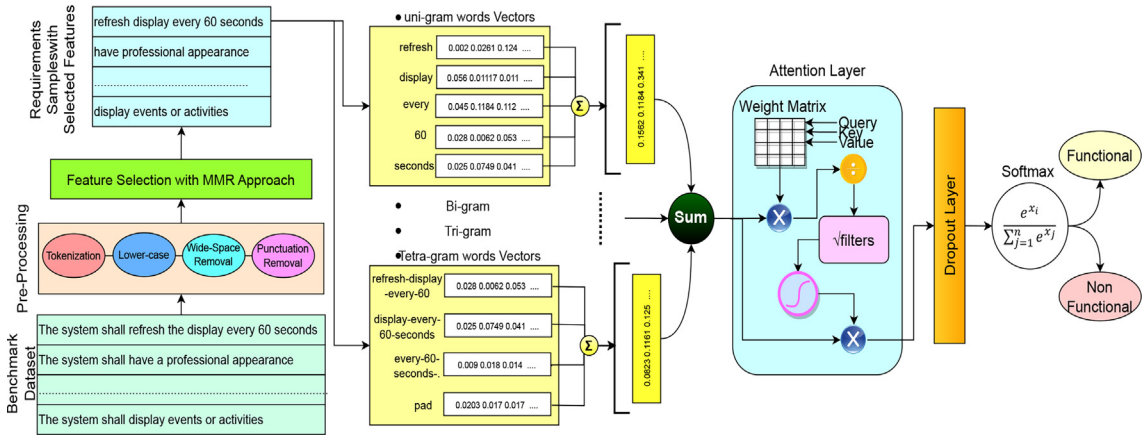

Fig. 4. Graphical illustration of proposed predictor workflow for requirements classification.

$$\text{Attention}[q,k,v] = \text{softmax}\frac{(q\cdot k^T)}{\sqrt{z_k}}v \tag{2}$$

Highly informative features are fed to dropout layer (Baldi and Sadowski, 2013) to avoid model overfitting. Finally, softmax classifier discriminates dense vectors into functional and non-functional classes.

3.4. Benchmark datasets

To evaluate the effectiveness of a variety of machine learning and deep leaning based methodologies for requirement classification, we have utilized two publicly available benchmark datasets; Promise (Sayyad Shirabad and Menzies, 2005) and Promise-exp (Lima et al., 2019). The majority of existing predictors are evaluated on these datasets (Khayashi et al., 2022; Tóth and Vidács, 2019). A comprehensive detail of these datasets is provided in existing papers, here we only summarize their statistics. The Promise dataset comprises 625 samples, whereas Promise-exp data consists of 970 samples of requirements. Table 3 provides a detail summary of Promise and Promise-exp datasets.

3.5. Evaluation measures

In order to assess the effectiveness of the proposed predictor for binary classification of requirements, a thorough comparative analysis of 19 existing predictors is conducted, which utilize different assessment metrics over two benchmark detests. The assessment metrics, which are widely used to examine the effectiveness of majority predictors, include precision, recall and F1-score. These matrices utilize true positive (t_p), true negative (t_n), false positive (f_p) and false negative (f_n) values to compute the predictor's overall performance. Mathematical formulation of precision, recall and F1-score is provided in Eqs. (3)–(5), respectively.

$$\text{Precision} = \frac{t_p}{t_p + f_p} \tag{3}$$

$$\text{Recall} = \frac{t_p}{t_p + f_n} \tag{4}$$

$$\text{F1} - \text{Score} = \frac{t_p}{t_p + \frac{(f_p + f_n)}{2}} \tag{5}$$

4. Experimental setup

We implemented adapted and proposed predictors on top of 7 APIs namely; keras [1], scikit-learn [2], numpy [3], math [4], scipy [5], pandas [6] and TensorFlow [7]. We have computed performance of machine learning classifiers using their default hyperparameters (MacKay, 1993). Furthermore, for proposed and adapted deep learning predictors the search space of different hyperparameters along with selected optimal values are provided in Table 4. Moreover, we have utilized Adam optimizer and softmax cross entropy loss for proposed and adapted deep learning predictors. Pre-trained Fasttext embed-

[1] https://keras.io/.
[2] https://scikit-learn.org/.
[3] https://numpy.org/.
[4] https://docs.python.org/3/library/math.html.
[5] https://scipy.org/.
[6] https://pandas.pydata.org/.
[7] https://www.tensorflow.org/.

Table 3
Statistical summary of benchmark datasets.

Dataset	Class	No. of Samples	Total Vocabulary	Unique Vocabulary	Sample Max_length	Sample Min_length	Sample Average length
Promise	Functional	255	1026	570	93	6	19
	Non-Functional	371	1750	1294	76	1	20
Promise-exp	Functional	444	1435	797	92	4	16
	Non-Functional	526	2241	1603	86	1	19

Table 4
Search space and selected values of hyperparameters for proposed and adapted deep learning predictors.

Hyperparameters	Search space	Selected
kernal Size	2,3,4,5,6,7	2,3,4
No. of kernels	8,16,32,64,100,128	8,16,100
Top_k_max_pooling	1,2,3,4,5,6,7,8,9,10	1,8
Hidden cells	3,4,5,6,7,8,9,10, 16, 32, 64,	5,9,64
Dropout_hidden_cell	0.1,0.01,0.001	0.1
Batch size	8,16,32,64	16
Learning rate	0.1,0.01,0.001,0.005,0.008,0.0001	0.008
Weight decay rate	0.1,0.5,1	1
Dropout_hidden_layer	0.01,0.001,0.001,0.0005,0.00001	0.0005
Random Embedding dimension	100	100
Adadelta_decay_rate	0.9,0.95,1	0.95
Adadelta_epsilon	$1\times e^{-5}$, $1\times e^{-6}$,$1 \times e^{-7}$,$1 \times e^{-8}$,$1 \times e^{-9}$	$1 \times e^{-8}$

dings (Khasanah, 2021) are used for feature representation apart from 2^{nd} layer of Yin et al. (2017) predictor and 2^{nd} and 3^{rd} layers of Zhang et al. (2016b) predictor which uses random embeddings.

5. Results

First, this section highlights the impact of 7 filter-based feature selection approaches on predictive performance of 11 machine learning classifiers. Further, it summarizes performance of 9 adapted and proposed deep learning predictors by using all input features and feature selection method based selected more informative features. It also presents a fair performance comparison of proposed predictor with best performing predictors from adapted machine and deep learning predictors over two public benchmark datasets using 6 different evaluation measures. Further, it provides an in-depth performance analysis of proposed and state-of-the-art requirements classification predictors over two benchmark datasets.

5.1. Hypotheses 1 and 2: Performance analysis of feature selection methods and machine learning classifiers

With an aim to validate Hypotheses 1 and 2, an in-depth performance comparison of 7 filter-based feature selection algorithms (ACC2 (Forman et al., 2003), BNS (Forman et al., 2003), CHISQ (Resnik, 1999), GINI (Park et al., 2010), MMR (Rehman et al., 2018), NDM (Rehman et al., 2017), POISON (Gao and Wang, 2009)) along with 11 machine learning classifiers (AdaBoost (Margineantu and Dietterich, 1997), Decision Tree (Quinlan, 1996), Extra Tree (Geurts et al., 2006), Gaussian NB (Leung et al., 2007), Gradient Boosting (Friedman, 2021), KNeighbour (Aha, 1997), Logistic Regression (LaValley, 2008), Naive Bayes (Leung et al., 2007), Random Forest (Breiman, 2001), Support Vector Machine (Hearst et al., 1998), Extreme Gradient Boosting (Chen et al., 2015)) at 24 different pre-defined benchmark test points of top-ranked features (100, 150, 200, 250, 300, 350, 400, 450, 500, 600, 650, 700, 750, 800, 850, 900, 950, 1000, 1050, 1100, 1150, 1200, 1250, 1300) over two benchmark datasets Promise and Promise-exp is provided in supplementary file Tables 1 and 2. A bird's-eye view of supplementary Tables 1 and 2 reveals that over both benchmark datasets, among 11 machine learning classifiers 3 classifiers namely; SVM, NB and LR produce highest performance values by using informative features selected through 5 different feature selection methods: MMR (Rehman et al., 2018), ACC2 (Forman et al., 2003), CHISQ (Resnik, 1999), POISON (Gao and Wang, 2009) and GINI (Park et al., 2010). On the other hand, 2 feature selection methods BNS (Forman et al., 2003) and NDM (Rehman et al., 2017) remain least performer in boosting the performance of machine learning classifiers. Similarly, 3 classifiers KNN, XGB and DT produce least performance while 4 classifiers (AdaBoost, ET, GNB and GB) performance remain in line between the predictive performance of top-performing (SVM, NB and LR) and least performing (KNN, XGB and DT) classifiers.

Table 5 illustrates performance values of 3 best performing machine learning classifiers, along with 5 top-performing feature selection methods at 7 predefined test points. Overall, it can be seen from Table 5, all 3 classifiers produce better performance when they are fed with discriminative features as compared to their baseline performance when they are fed with all input features. Furthermore, as compared to baseline with selected features, there is a slight performance gain. This primarily because requirements samples contain short sentences and we observe feature selection is more comprehensive when samples contain a large body of content. However, although samples contain short content, still there is slight performance gain.

To obtain appropriate performance of classifier, selection of top-ranked features is important. It can be seen in Table 5 over Promise dataset, all feature selection algorithms along with 3 classifiers produce better performance at top 500 and 750 features. Similarly, in Promise-exp dataset, SVM and NB produce better performance with top 750 features and LR produces better performance with

Table 5
Performance comparison of 5 feature selection algorithms at 7 benchmark test points using 3 machine learning classifiers over Promise and Promise-exp datasets.

		Benchmark Test Points Promise Dataset								Benchmark Test points Promise_Expanded Dataset							
classifier	FS_algo	100	200	500	750	1000	1200	1300	all	100	200	500	750	1000	1200	1300	all
Logistic Regression	ACC2	0.773	0.842	0.858	0.858	0.845	0.845	0.845		0.741	0.814	0.845	0.856	0.858	0.809	0.809	
	CHISQ	0.766	0.840	0.861	0.860	0.845	0.845	0.845		0.739	0.819	0.844	0.856	0.858	0.809	0.809	
	GINI	0.773	0.842	0.858	0.858	0.845	0.845	0.845		0.741	0.814	0.845	0.856	0.858	0.809	0.809	
	MMR	0.773	0.842	0.858	0.858	0.845	0.845	0.845		0.741	0.814	0.845	0.856	0.858	0.809	0.809	
	POISON	0.773	0.842	0.858	0.858	0.845	0.845	0.845		0.741	0.814	0.845	0.856	0.858	0.809	0.809	
	Baseline								0.811								0.823
Naive_Bayes	ACC2	0.759	0.815	0.855	0.869	0.817	0.817	0.817		0.741	0.802	0.841	0.869	0.869	0.804	0.804	
	CHISQ	0.751	0.817	0.855	0.875	0.817	0.817	0.817		0.742	0.804	0.841	0.870	0.870	0.804	0.804	
	GINI	0.759	0.815	0.855	0.869	0.817	0.817	0.817		0.741	0.802	0.841	0.869	0.869	0.804	0.804	
	MMR	0.759	0.815	0.855	0.869	0.817	0.817	0.817		0.741	0.802	0.841	0.869	0.869	0.804	0.804	
	POISON	0.759	0.815	0.855	0.869	0.817	0.817	0.817		0.741	0.802	0.841	0.869	0.869	0.804	0.804	
	Baseline								0.826								0.835
Support Vector Machine	ACC2	0.766	0.845	0.864	0.872	0.827	0.827	0.827		0.726	0.801	0.838	0.878	0.875	0.816	0.816	
	CHISQ	0.759	0.836	0.864	0.873	0.827	0.827	0.827		0.725	0.806	0.837	0.878	0.876	0.816	0.816	
	GINI	0.766	0.845	0.864	0.872	0.827	0.827	0.827		0.726	0.801	0.838	0.878	0.875	0.816	0.816	
	MMR	0.766	0.845	0.864	0.872	0.827	0.827	0.827		0.726	0.801	0.838	0.878	0.875	0.816	0.816	
	POISON	0.766	0.845	0.864	0.872	0.827	0.827	0.827		0.726	0.801	0.838	0.878	0.875	0.816	0.816	
	Baseline								0.819								0.803

top 1000 features. When we select only a few number of features in this particular scenario, some samples which do not contain selected features become empty and predictive pipeline randomly puts an unknown word in those samples and classifier makes random guess, this deteriorates the performance of predictive pipeline. Conversely, when we select a large set of top-ranked features, it may still contain some irrelevant and redundant features that also affect performance of predictive pipeline.

Furthermore, overall different feature selection methods along with different classifers produce distinct performance values at different test points. However, overall feature selection methods produce almost similar performance values. This is primarily because requirements samples contain short snippets of text and when we select less number of top ranked features most of the samples become empty. Contrarily, when we select large number of features than most of the methods contain same number of top-ranked features and they produce almost similar performance values.

5.2. Hypotheses 3 and 4: Performance analysis of adapted and proposed deep learning predictors

After investigating performance of machine learning classifiers with 7 different feature engineering approaches, we have chosen the best-performing feature selection approach (MMR) (Rehman et al., 2018) to validate Hypotheses 3 and 4 by analyzing the impact of selected features on predictive performance of proposed and adapted deep learning predictors. Detailed performance figures of proposed and adapted deep learning predictors over 10 pre-defined benchmark test points of top-ranked features (100, 200, 300, 400, 500, 600, 700, 800, 900, 1000) in terms of F1-score is provided in the supplementary file in Tables 4, 5 and 6. A closer look at supplementary Tables 4, 5 and 6 reveals that, similar to machine learning based classifiers, here once again deep learning predictors produce different performance values at different test points. Fig. 5 graphically illustrates performance of proposed and adapted deep learning predictors by feeding complete features and top-ranked features selected through MMR approach (Rehman et al., 2018) at best benchmark test points over Promise and Promise-exp datasets, respectively. Here, the word baseline refers to predictors with complete features.

Considering working paradigm among adapted deep learning predictors 4 ((Kim, 2014; Kalchbrenner et al., 2014; Yin et al., 2017; Zhang et al., 2016b)) belongs CNN-based architecture, 2 ((Yogatama et al., 2017; Palangi et al., 2016)) have RNN-based architecture and 3 predictors ((Chen et al., 2017; Zhou et al., 2015; Wang et al., 2016)) have hybrid architecture. Fig. 5 (A) depicts F1-scores of adapted and proposed deep learning predictors using all input features and selected features of Promise dataset. Among all predictors, RNN-based (Palangi et al., 2016; Yogatama et al., 2017) baseline predictors manage to achieve the highest F1-score of 0.691 and 0.599, respectively. Among all adapted predictors, group of CNN-based baseline predictors remains worse performing by attaining minimal F1-Score. Primarly CNN based architectures extracted discriminative features and remains fail in extracting semantic information about words and produced poor performance in comparison to RNN based architectures which managed to capture context of requirements samples. However, it is evident from Fig. 5 that utilizing only discriminative features elected through MMR approach (Rehman et al., 2018) remarkably rises performance of all adapted deep learning predictors. Performance of 5 deep learning predictors ((Kalchbrenner et al., 2014; Yin et al., 2017; Zhang et al., 2016b; Wang et al., 2016; Chen et al., 2017)) effectively improves with an increase of approximately 10% in F1-score by using 500, 200, 800, 700 and 800 features, respectively. Particularly, (Zhang et al., 2016b) CNN-based predictor observes a performance boost of around 18% by utilizing top-ranked discriminate features in comparison to baseline predictor. Proposed predictor, which utilizes self-attention (Vaswani et al., 2017) based deep learning predictor, drastically improves F1-Score from 0.936 to 0.945 by utilizing discriminative features identified by filter-based feature selection algorithm (MMR) (Rehman et al., 2018) instead of using complete features. Along with deep learning predictors,iInduction of feature selection method reveals that when predictors have comprehensive discriminative features than even if they do not capture semantical information still they can produce better performance.

It can be inferred from Fig. 5 (B) over Promise-exp dataset baseline of adapted deep learning predictors exhibit a discouraging performance with F1-Score of approximately 50% over Promise-exp dataset. Kalchbrenner et al. (2014) CNN-based predictor is the worst performer with F1-score of 0.42%. Conversely, instead of complete features, feeding deep learning predictors with the most relevant features results in a considerable rise in efficiency overall deep learning predictors. F1-score of 6 deep learning predictors including; (Kalchbrenner et al., 2014; Yogatama et al., 2017; Kim, 2014; Zhang et al., 2016b; Wang et al., 2016; Zhou et al., 2015) predictors rise roughly 10% by selecting top-ranked features. The

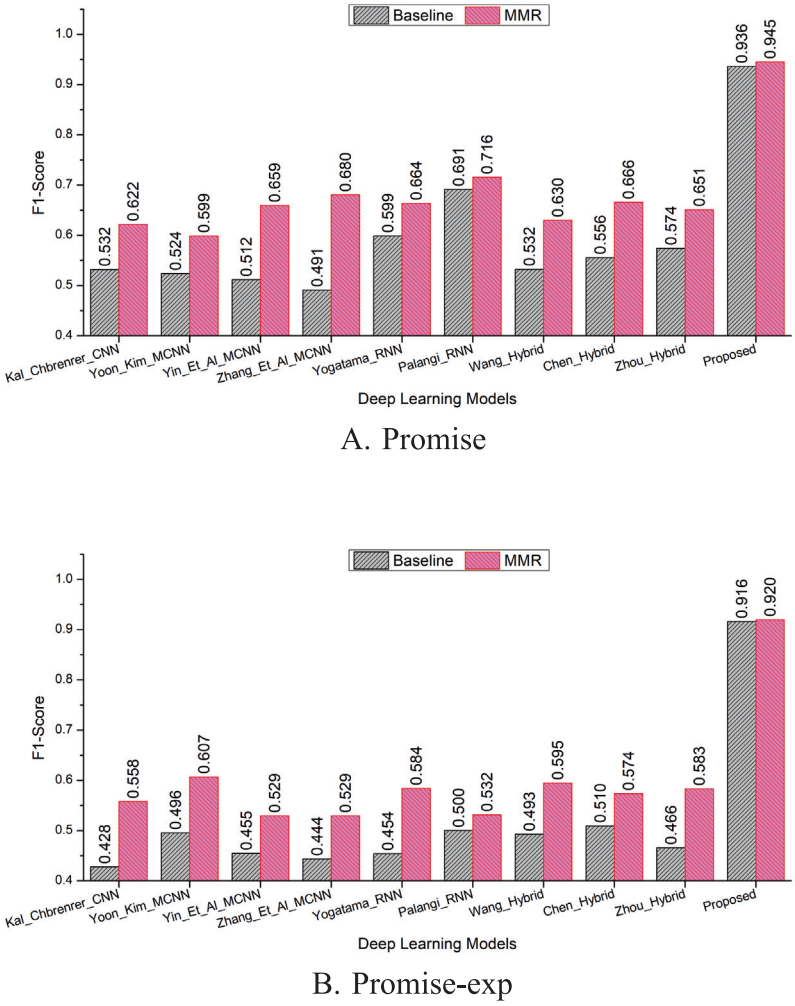

Fig. 5. Performance comparison of best feature set of MMR algorithm with baseline using 9 adapted deep learning predictors over benchmark datasets.

remarkable performance increase in adapted deep learning predictors over benchmark datasets by feeding features selected through Max–Min ratio (MMR) (Rehman et al., 2018), highlights the potential of selected discriminative features for more accurate predictions. The proposed predictor surpasses the performance of the best performing deep learning predictor by performance gain of 30% and 32% with complete and selected features, respectively. Performance of the proposed predictor improves with a margin of 1% by utilizing selected features as compared to baseline predictor relying on complete features.

In a nutshell, proposed and adapted deep learning predictors significantly produce better performance when they are fed with features selected through filter-based feature selection algorithm (MMR) (Rehman et al., 2018) over both datasets. Apart from predictive performance, incorporation of feature selection method in the predictive pipeline also reduces the computational complexity of deep learning predictors. MMR approach (Rehman et al., 2018) eliminates noisy and irrelevant features in the underlay corpus and retains only the most informative features that facilitate predictors in accurately classifying requirements into functional and non-functional classes. Furthermore, performance over both datasets reveals that, a more powerful classification predictor can be designed by focusing on discriminative features. Specifically, proposed predictor produces state-of-the-art performance values on both benchmark datasets as it reaps the benefits of traditional feature selection method and in the predictor attention layer (Vaswani et al., 2017) also focuses on more discriminative features.

5.3. Performance comparison of proposed predictor with top-performing adapted machine learning and deep learning predictors

A large-scale experimental analysis of adapted machine learning predictive pipelines reveals that among all machine learning classifiers, SVM classifier along with MMR feature selection method (Rehman et al., 2018) manages to produce highest performance over two benchmark datasets. On the other hand, from adapted deep learning predictors optimal combination of MMR feature selection method (Rehman et al., 2018) and best performing predictor varies according to dataset such as (Palangi et al., 2016 and Kim, 2014) predictors produce better performance over Promise and Promise-exp datasets, respectively. This section illustrates performance comparison of proposed and top-performing adapted machine and deep learning predictors (Palangi et al., 2016; Kim, 2014) using 6 different evaluation measures including accuracy, precision, recall, F1-score, sensitivity and specificity.

It is evident from Fig. 6 that among adapted machine and deep learning predictors, MMR feature selection method (Rehman et al., 2018) and SVM classifier based predictive pipeline surpasses performance of (Palangi et al., 2016 and Kim, 2014) predictors with significant margin of around 15% and 20% across all evaluation

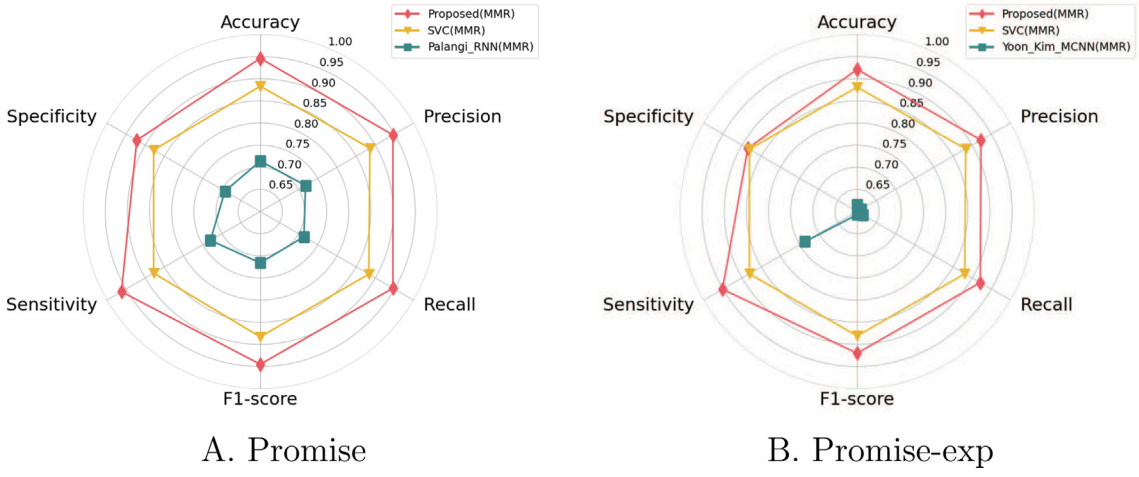

Fig. 6. Performance comparison of proposed and top-performing adapted machine and deep learning predictors over benchmark datasets.

Table 6
Performance comparison of proposed and existing predictors over Promise dataset under independent test set.

Author	Predictor	F1-Score	Precision	Recall
(Tiun et al., 2020)	BoW + LR	89.26	88.72	90.82
	BoW + NB	91.8	62.76	91.05
	BoW + SVM	61.17	91.4	84.57
	TFIDF + LR	92.41	91.99	92.96
	TFIDF + NB	91.13	93.48	89.78
	TFIDF + SVM	38.42	31.2	50
	Doc2Vec + LR	82.66	82.25	84
	Doc2Vec + NB	82.55	82.11	83.58
	Doc2Vec + SVM	73.47	83.46	72.12
	Lexical Features + SVM	92	92	92
	Word2Vec + CNN	92.16	92.68	91.87
	FastText	92.8	92.8	92.8
-	**Proposed**	**96.8**	**96.8**	**96.8**

measures over both promise and promise-exp datasets under 10-fold cross-validation experimental setting. It is widely accepted that deep learning predictors perform better on large datasets and machine learning predictors even produce better performance on small datasets. Both requirements classification datasets have fewer samples that hinder predictive performance of deep learning predictors. On the other hand, proposed predictor when fed with discriminative feature set outperforms SVM classifier with a margin of about 5% across both datasets in terms of all evaluation matrices. Furthermore, proposed predictor outperforms deep learning predictors (Palangi et al., 2016; Kim, 2014) with performance gain of approximately 20% and 25% percent over promise and promise-exp datasets, respectively.

Although, proposed predictor is also deep leaning based but unlike adapted deep learning predictors, it does not make use of any CNN (O'Shea and Nash, 2015) or RNN architecture and utilizes self-attention mechanism (Vaswani et al., 2017) that empowers focus on more informative features. Thus, proposed predictor exhibits dominating performance over both machine learning and deep learning predictors using both benchmark datasets of requirements classification.

5.4. Performance comparison of proposed and existing requirements classification predictors

This section compares performance of proposed and existing requirements classification predictors over two benchmark datasets (Promise and Promise-exp) in terms of 3 different evaluation measures namely precision, recall and F1-score. Over, Promise-exp dataset existing predictors are evaluated under 10-fold cross-validation setting while on Promise dataset existing predictors are evaluated in two different experimental settings: 10-fold cross-validation and independent test set with 80–20 ratio. To perform a fair performance comparison, we compare the performance of proposed predictor with existing predictors using 10-fold experimental setting on Promise-exp dataset and both 10-fold cross-validation as well as independent test set based experimental settings on Promise dataset.

5.4.1. Proposed and existing predictors performance comparison over Promise dataset under independent test set experimental setting

Table 6 compares performance of proposed and existing predictors over Promise independent test set. Among existing machine learning requirements classification predictive pipelines (Tiun et al., 2020), NB classifier with bag of words (BoW) based feature representation approach produces better performance as compared to other two classifiers namely SVM and LR. In comparison to BoW based approach, TFIDF feature representation approach (Ramos et al., 2003) boosts the performance of LR classifier. Primarily, this is because BoW based approach transforms requirements into statistical feature space by computing term frequencies of words, while TFIDF approach (Ramos et al., 2003) makes use of both term frequencies and document frequencies of words and assigns more comprehensive scores to informative features. Unlike LR, NB classifier produces almost similar performance with both BoW and TFIDF approaches. Unexpectedly, performance

Table 7
Performance comparison of proposed and existing predictors over Promise 10-fold cross-validation.

Author	Predictor	F1-Score	Precision	Recall
(Kurtanović and Maalej, 2017)	Lexical features + SVM	92.5	92.5	92.5
(Abad et al., 2017)	Processed data	94	94	95
(Hey et al., 2020)	NoRBERT (large, ep.=10, OS)			
	91.5	92	91.5	
(Li et al., 2022)	DBGAT	94	94	94
-	**Proposed**	**94.6**	**94.6**	**94.5**

Table 8
Performance comparison of proposed and baseline predictors for binary classification of requirements over Promise-exp 10-fold cross-validation.

Author	Predictor	F1-Score	Precision	Recall
(Dias Canedo and Cordeiro Mendes, 2020)	KNN	82	82	82
	SVM	91	91	91
-	**Proposed**	**92**	**92**	**92**

of svm classifier decreases with TFIDF representation (Ramos et al., 2003) as compared to its performance with simple BoW based representation approach (Zhang et al., 2010).

Contrarily, Doc2Vec approach that makes use of embedding generation method along with NB and LR classifiers remains fail to outperform these classifiers with BoW (Zhang et al., 2010) and TFIDF (Ramos et al., 2003) based feature representation approaches. However, along with this feature representation approach, SVM classifier produces better performance as compared to its performance with BoW (Zhang et al., 2010) and TFIDF (Ramos et al., 2003) based feature representation methods.

Word2vec and FastText embedding generation methods based statistical representation along with CNN (O'Shea and Nash, 2015) and linear classifiers produce almost similar performance by achieving 92.16% and 92.8% F1-scores. SVM classifier along with lexical features where authors first tagged requirements text with adjectives nouns, verbs and adverbs produces 92% F1-score. Overall, it can be concluded in existing approaches feature representation methods showed important role in the predictive performance of machine and deep learning classifiers. Among all feature representation approaches, FastText approach produced highest performance as it generates statistical representation by considering subwords based distribution of content and is more competent in extracting semantics of text.

On the other hand, proposed approach that makes use of selected more discriminative features and attention-based mechanism outperforms all existing approaches and produces highest performance by achieving F1-score 96.8%. Overall, based on experimental results it can be concluded, in requirements classification predictive pipelines feature representation part is more important to achieve better performance. However, predictive performance of proposed approach reveals that as compared to feature representation part, if predictive pipeline is enriched with feature selection and attention architecture, even along with random embeddings predictive pipeline can produce decent performance. We believe along with pretrained word embeddings predictive performance of proposed predictor can further be improved.

5.4.2. Proposed and existing predictors performance comparison over Promise dataset under 10-fold cross-validation experimental setting

Table 7 illustrates performance comparison of proposed and existing predictors over Promise dataset under 10-fold cross-validation. Among existing predictors, Bert language model (Hey et al., 2020) produces least performance by achieving 91.5% F1-score. Although, language models have produced state-of-the-art performance for diverse types of natural language processing applications including textual document classification (Liao et al., 2023), sentiment analysis (Alaparthi and Mishra, 2021), hate speech detection (Mozafari et al., 2019) and question answering systems (Qu et al., 2019). Mainly, language models working paradigm can be summarized into 3 main steps, unsupervised training on large generic textual data and fine-tuning on domain-specific text. Finally, supervised training and evaluation of model. Specifically, in requirements classification tasks, domain-specific data is very less and it cannot be properly utilized to fine-tune language models which is why BERT based predictor remain fail to produce better performance. Kurtanović and Maalej (2017) SVM based predictor produces performance even better than Bert language model. This is primarily because authors (Kurtanović and Maalej, 2017) enriched predictor with lexical features, such as by first tagging textual data with nouns, adjectives and adverbs. The induction of lexical features enhances the performance of svm classifier. Abad et al. (2017) and Li et al. (2022) predictors produce 94% F1-score that is better than both svm and Bert based language models. However, Abad et al. (2017) predictor is not generic as it categorizes requirements into functional and non-functional classes based on predefined rules only. Predefined rules are accurately categorizing requirements into predefined classes but probably these rules may not categorize requirements in real-time deployment. Overall, among existing approaches, (Li et al., 2022) predictor is more accurate as it makes use of a graph-based embedding generation strategy along with Bert language model.

5.4.3. Proposed and existing predictors performance comparison over Promise-exp dataset under 10-fold cross-validation experimental setting

Table 8 highlights performance comparison of proposed and existing predictors for discriminating functional and non-functional requirements over Promise-exp dataset using 10-fold cross-validation experimental setting. Dias Canedo and Cordeiro Mendes (2020)) approach investigates performance of BoW based feature representation with traditional machine learning classifiers namely; KNN and SVM. SVM outperforms KNN with performance gain of 9% across all 3 measures precision, recall and F1-Score. Proposed predictor based on feature selection method and attention mechanism once again outperforms (Dias Canedo and Cordeiro Mendes, 2020) KNN predictor by 10% and SVM predictor by 1% across all 3 evaluation measures. Here it can be seen from Table 8, all predictors have same precision and recall scores, which illustrates that both existing and proposed predictors are robust as they are neither biased towards type I or type II errors.

6. Discussion

- Hypothesis 1: Do traditional feature selection methods improve machine learning classifier's performance?
  In traditional machine learning based predictive pipelines feature selection becomes essential when number of features becomes larger than number of samples. However, in current problem number of features are not greater than number of samples and theoretically it is not necessary to perform feature selection. Experimental results reveal that over-requirement classification datasets feature selection has improved the performance of predictors. This means even when number of features is less than number of samples still feature selection can slightly improve classification performance. On the other hand, in a particular task in which number of features is greater than number of samples, feature selection boosts more performance as compared to the performance it has boosted in current task.
- Which optimal combination of feature selection algorithm and classifier produces better performance?.
  Min–Max Ratio (MMR) feature selection method along with SVM and LR produces better performance based on different benchmark test points classifier. Primarily, MMR feature selection method assigns higher scores to more discriminative words and lower scores to less discriminative as well as frequent and rare words.
- Dose combined potential of traditional feature selection and attention architecture enhance classifier performance by focusing on most informative features?.
  Our prime objective was to analyze, performance behavior of deep learning predictors when they are fed with all input features and with only the most discriminative features. We noticed that when deep learning predictors were fed with input samples having all features they produce comparably less performance as compared to their performance when they were fed with samples having the most discriminative features. Specifically, in the first scenario, we feed deep learning models with samples having all features in them. In the second scenario, we utilize a traditional feature selection method to remove irrelevant and redundant features from input samples (Tutsoy et al., 2020). Experimental results reveal that external feature engineering improves their predictive performance and reduces their computational complexity.
- Dose standalone attention mechanism based classifier produce performance better than CNN and RNN architecture based classifiers?.
  Deep learning predictors require large training data and in requirement classification task training data is small. That is why traditional deep learning predictors produce less performance. According to nature of the requirements data, we designed a lightweight deep learning predictor that only makes use of the attention mechanism and does not use any CNN or RNN layers.

7. Limitations of study

Mainly requirement classification falls into two different categories namely binary and multi-class. In binary classification, requirements are categorized into functional and non-functional classes while in multi-class classification, requirements are categorized into functional or nonfunctional subclasses. In binary classification, there exist comprehensive discriminative features between functional and nonfunctional classes and based on discriminative feature requirements can be accurately categorized. However, in case of multi-class classification, there exist less discriminative patterns among different non-functional classes. Therefore, to precisely categorize requirements into different types of non-functional classes machine learning algorithms require both discriminative as well as contextual information. The proposed predictor focuses on discriminative features so it can be utilized for binary class classification and it may not perform better for multi-class classification. However, for multi-class classification, its predictive performance can be improved by incorporating additional branches that focus on the extraction of contextual information.

Prime focus of this study was to analyze the impact of traditional feature selection algorithms on the predictive performance of classifiers. However, apart from suitable subset of features, performance of traditional machine learning classifiers also relies on the optimal combination of their hyperparameters values. In this study, experimentation is performed with classifiers' default hyperparameters. Along with selected informative subsets of features, selection of classifiers' optimal hyperparameters values may further improve classifiers' predictive performance. Additionally, in this study, small size datasets are used and there is need for development of large benchmark datasets for the development of more powerful and generic predictors.

8. Conclusion

Accurate discrimination of requirements into functional and non-functional classes is a fundamental step of requirement engineering. In existing predictors, focus of researchers has been on exploring better feature representation approaches for the development of better predictive pipelines competent in precisely categorizing requirements into functional and non-functional classes. In existing approaches, potential of feature selection approaches remains unexplored. The paper in hand explores combined potential of 7 filter-based feature selection algorithms and 11 machine learning classifiers for requirements classification over two benchmark datasets. It also presents a unique study, to simultaneously reap the benefits of traditional feature selection and diverse types of deep learning predictors. Experimental results reveal that traditional feature selection approaches boost the perdictive performance of both machine and deep learning predictors. Moreover, this paper presents a novel predictor that utilizes power of traditional feature selection techniques with self-attention based linear classifier. The proposed predictor outperforms state-of-the-art requirements classification approaches with a significant margin of 4% and 1% over Promise and Promise-exp datasets in terms of F1-Score. We believe proposed predictor performance can be further improved by incorporating pretrained word embeddings.

To further improve the predictive performance of requirements classification, a compiling direction of work presented in this paper will be to explore the potential of different word embedding methods (Word2Vec, FastText, Node2Vec, GLoVE, LINE, etc.). In current study deep learning predictors make use of randomly initialized embeddings. However, requirements classification datasets are small in size and utilization of pre-trained word embeddings may enhance the performance of deep learning predictors. Primarily word embeddings learn distributions of words in unsupervised fashion and utilization of pre-trained vectors of words at embedding layers of deep learning classifiers facilitate them to produce better performance even on small size datasets. Another possible direction is to find optimal hyperparameters of traditional machine learning classifiers by utilizing any genetic algorithm.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Supplementary material

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.jksuci.2023.101665.

References

Abad, Z.S.H., Karras, O., Ghazi, P., Glinz, M., Ruhe, G., Schneider, K., 2017. What works better? a study of classifying requirements. 2017 IEEE 25th International Requirements Engineering Conference (RE). IEEE, pp. 496–501.

Abbasi, M.S., Al-Sahaf, H., Mansoori, M., Welch, I., 2022. Behavior-based ransomware classification: A particle swarm optimization wrapper-based approach for feature selection. Appl. Soft Comput. 121, 108744.

Aha, D., 1997. Special ai review issue on lazy learning. Artif. Intell. Rev. 11.

Ajagbe, M., Zhao, L., 2022. Retraining a bert model for transfer learning in requirements engineering: A preliminary study. 2022 IEEE 30th International Requirements Engineering Conference (RE). IEEE, pp. 309–315.

Alaparthi, S., Mishra, M., 2021. Bert: A sentiment analysis odyssey. J. Market. Anal. 9 (2), 118–126.

Althanoon, A.A.A., Younis, Y.S., 2021. Supporting classification of software requirements system using intelligent technologies algorithms.

Asim, M.N., Khan, M.U.G., Malik, M.I., Razzaque, K., Dengel, A., Ahmed, S., 2019. Two stream deep network for document image classification. 2019 International Conference on Document Analysis and Recognition (ICDAR). IEEE, pp. 1410–1416. https://doi.org/10.1109/ICDAR.2019.00227.

Asim, M.N., Ghani, M.U., Ibrahim, M.A., Mahmood, W., Dengel, A., Ahmed, S., 2021. Benchmarking performance of machine and deep learning-based methodologies for urdu text document classification. Neural Comput. Appl. 33, 5437–5469.

Baker, C., Deng, L., Chakraborty, S., Dehlinger, J., 2019. Automatic multi-class non-functional software requirements classification using neural networks. 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC). IEEE, pp. 610–615.

Balaji, S., Murugaiyan, M.S., 2012. Waterfall vs. v-model vs. agile: A comparative study on sdlc. Int. J. Informat. Technol. Bus. Manage. 2 (1), 26–30.

Baldi, P., Sadowski, P.J., 2013. Understanding dropout. Adv. Neural Informat. Process. Syst. 26.

Becker, P., Tebes, G., Peppino, D., Olsina Santos, L.A., 2019. Applying an improving strategy that embeds functional and non-functional requirements concepts. J. Comput. Sci. Technol. 19.

Breiman, L., 2001. Random forests. Mach. Learn. 45, 5–32.

Chaiani, M., Selouani, S.A., Boudraa, M., Yakoub, M.S., 2022. Voice disorder classification using speech enhancement and deep learning models. Biocybernet. Biomed. Eng. 42 (2), 463–480.

Chen, Y., 2015. Convolutional neural network for sentence classification. University of Waterloo. Master's thesis.

Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., Chen, K., Mitchell, R., Cano, I., Zhou, T., et al., 2015. Xgboost: extreme gradient boosting. R package version 0.4-2 1 (4), 1–4.

Chen, G., Ye, D., Xing, Z., Chen, J., Cambria, E., 2017. Ensemble application of convolutional and recurrent neural networks for multi-label text categorization. 2017 International Joint Conference on Neural Networks (IJCNN). IEEE, pp. 2377–2383.

Dias Canedo, E., Cordeiro Mendes, B., 2020. Software requirements classification using machine learning algorithms. Entropy 22 (9), 1057.

El Aboudi, N., Benhlima, L., 2016. Review on wrapper feature selection approaches. In: 2016 International Conference on Engineering & MIS (ICEMIS). IEEE, pp. 1–5.

Fávero, E.M.D.B., Casanova, D., 2021. Bert_se: A pre-trained language representation model for software engineering, arXiv preprint arXiv:2112.00699.

Felderer, M., Travassos, G.H., 2020. Contemporary Empirical Methods in Software Engineering. Springer.

Forman, G. et al., 2003. An extensive empirical study of feature selection metrics for text classification. J. Mach. Learn. Res. 3 (Mar), 1289–1305.

Friedman, J., 2021. Greedy boosting approximation: a gradient boosting machine. Annals Stat.

Gao, Y., Wang, H.-L., 2009. A feature selection algorithm based on poisson estimates. In: 2009 Sixth International Conference on Fuzzy Systems and Knowledge Discovery, vol. 1, IEEE, pp. 13–18.

Geurts, P., Ernst, D., Wehenkel, L., 2006. Extremely randomized trees. Mach. Learn. 63, 3–42.

Haghighat, A., Sharma, A., 2023. A computer vision-based deep learning model to detect wrong-way driving using pan–tilt–zoom traffic cameras. Comput.-Aided Civil Infrastruct. Eng. 38 (1), 119–132.

Haque, M.A., Rahman, M.A., Siddik, M.S., 2019. Non-functional requirements classification with feature extraction and machine learning: An empirical study. In: 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT). IEEE, pp. 1–5.

Hearst, M.A., Dumais, S.T., Osuna, E., Platt, J., Scholkopf, B., 1998. Support vector machines. IEEE Intell. Syst. Appl. 13 (4), 18–28.

Hey, T., Keim, J., Koziolek, A., Tichy, W.F., 2020. Norbert: Transfer learning for requirements classification. 2020 IEEE 28th International Requirements Engineering Conference (RE). IEEE, pp. 169–179.

Hidellaarachchi, D., Grundy, J., Hoda, R., Madampe, K., 2021. The effects of human aspects on the requirements engineering process: A systematic literature review. IEEE Trans. Software Eng.

Horkoff, J., 2019. Non-functional requirements for machine learning: Challenges and new directions. 2019 IEEE 27th International Requirements Engineering Conference (RE). IEEE, pp. 386–391.

Ivanov, V., Sadovykh, A., Naumchev, A.Bagnato, A., Yakovlev, K., 2022. Extracting software requirements from unstructured documents. In: Recent Trends in Analysis of Images, Social Networks and Texts: 10th International Conference, AIST 2021, Tbilisi, Georgia, December 16–18, 2021, Revised Selected Papers, Springer, pp. 17–29.

Jalal, N., Mehmood, A., Choi, G.S., Ashraf, I., 2022. A novel improved random forest for text classification using feature ranking and optimal number of trees. J. King Saud Univ.-Comput. Informat. Sci. 34 (6), 2733–2742.

Jarzebowicz, A., Weichbroth, P., 2021. A systematic literature review on implementing non-functional requirements in agile software development: Issues and facilitating practices. In: Lean and Agile Software Development: 5th International Conference, LASD 2021, Virtual Event, January 23, 2021, Proceedings 5, Springer, pp. 91–110.

Jović, A., Brkić, K., Bogunović, N., 2015. A review of feature selection methods with applications. 2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). Ieee, pp. 1200–1205.

Kalchbrenner, N., Grefenstette, E., Blunsom, P., 2014. A convolutional neural network for modelling sentences. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Baltimore, Maryland, pp. 655–665. https://doi.org/10.3115/v1/P14-1062. https://aclanthology.org/P14-1062.

Kaur, K., Kaur, P., 2022. Sabdm: A self-attention based bidirectional-rnn deep model for requirements classification. J. Softw.: Evol. Process, e2430.

Khamparia, A., Singh, P.K., Rani, P., Samanta, D., Khanna, A., Bhushan, B., 2021. An internet of health things-driven deep learning framework for detection and classification of skin cancer using transfer learning. Trans. Emerg. Telecommun. Technol. 32 (7), e3963.

Khasanah, I.N., 2021. Sentiment classification using fasttext embedding and deep learning model. Proc. Comput. Sci. 189, 343–350.

Khayashi, F., Jamasb, B., Akbari, R., Shamsinejadbabaki, P., 2022. Deep learning methods for software requirement classification: A performance study on the pure dataset. arXiv preprint arXiv:2211.05286.

Khurana, D., Koli, A., Khatter, K., Singh, S., 2023. Natural language processing: State of the art, current trends and challenges. Multimedia Tools Appl. 82 (3), 3713–3744.

Kici, D., Bozanta, A., Cevik, M., Parikh, D., Başar, A., 2021a. Text classification on software requirements specifications using transformer models. In: Proceedings of the 31st Annual International Conference on Computer Science and Software Engineering, pp. 163–172.

Kici, D., Malik, G., Cevik, M., Parikh, D., Basar, A., 2021b. A bert-based transfer learning approach to text classification on software requirements specifications. In: Canadian Conference on AI.

Kim, Y., 2014. Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Doha, Qatar, pp. 1746–1751. https://doi.org/10.3115/v1/D14-1181. https://aclanthology.org/D14-1181.

Knauss, E., Houmb, S., Schneider, K., Islam, S., Jürjens, J., 2011. Supporting requirements engineers in recognising security issues. In: Requirements Engineering: Foundation for Software Quality: 17th International Working Conference, REFSQ 2011, Essen, Germany, March 28–30, 2011. Proceedings 17, Springer, pp. 4–18.

Kumar, A., Jaiswal, A., Garg, S., Verma, S., Kumar, S., 2022. Sentiment analysis using cuckoo search for optimized feature selection on kaggle tweets. In: Research Anthology on Implementing Sentiment Analysis Across Multiple Disciplines, IGI Global, pp. 1203–1218.

Kurtanović, Z., Maalej, W., 2017. Automatically classifying functional and non-functional requirements using supervised machine learning. 2017 IEEE 25th International Requirements Engineering Conference (RE). Ieee, pp. 490–495.

LaValley, M.P., 2008. Logistic regression. Circulation 117 (18), 2395–2399.

Leclercq, M., Vittrant, B., Martin-Magniette, M.L., Scott Boyer, M.P., Perin, O., Bergeron, A., Fradet, Y., Droit, A., 2019. Large-scale automatic feature selection for biomarker discovery in high-dimensional omics data. Front. Genet. 10, 452.

Leung, K.M. et al., 2007. Naive bayesian classifier. Polytechnic Univ. Depart. Comput. Sci./Finance Risk Eng. 2007, 123–156.

Liao, W., Liu, Z., Dai, H., Wu, Z., Zhang, Y., Huang, X., Chen, Y., Jiang, X., Zhu, D., Liu, T. et al., 2023. Mask-guided bert for few shot text classification, arXiv preprint arXiv:2302.10447.

Li, G., Zheng, C., Li, M., Wang, H., 2022. Automatic requirements classification based on graph attention network. IEEE Access 10, 30080–30090.

Lima, M., Valle, V., Costa, E., Lira, F., Gadelha, B., 2019. Software engineering repositories: Expanding the promise database. In: Proceedings of the XXXIII Brazilian Symposium on Software Engineering, pp. 427–436.
Lualdi, M., Fasano, M., 2019. Statistical analysis of proteomics data: a review on feature selection. J. Proteomics 198, 18–26.
Luo, X., Xue, Y., Xing, Z., Sun, J., 2022. Prcbert: Prompt learning for requirement classification using bert-based pretrained language models. In: 37th IEEE/ACM International Conference on Automated Software Engineering, pp. 1–13.
MacKay, D.J., 1993. Hyperparameters: optimize, or integrate out? Maximum Entropy Bayesian Methods: Santa Barbara, California, USA 1996, 43–59.
Maldonado, S., López, J., 2018. Dealing with high-dimensional class-imbalanced datasets: Embedded feature selection for svm classification. Appl. Soft Comput. 67, 94–105.
Margineantu, D.D., Dietterich, T.G., 1997. Pruning adaptive boosting. ICML, vol. 97. Citeseer, pp. 211–218.
Maruping, L.M., Matook, S., 2020. The evolution of software development orchestration: current state and an agenda for future research. Eur. J. Informat. Syst. 29 (5), 443–457.
Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., Gao, J., 2021. Deep learning–based text classification: a comprehensive review. ACM Comput. Surv. (CSUR) 54 (3), 1–40.
Mozafari, M., Farahbakhsh, R., Crespi, N., 2019. A bert-based transfer learning approach for hate speech detection in online social media. In: Complex Networks and Their Applications VIII: Volume 1 Proceedings of the Eighth International Conference on Complex Networks and Their Applications COMPLEX NETWORKS 2019 8, Springer, pp. 928–940.
Onan, A., 2021. Sentiment analysis on massive open online course evaluations: a text mining and deep learning approach. Comput. Appl. Eng. Educ. 29 (3), 572–589.
O'Shea, K., Nash, R., 2015. An introduction to convolutional neural networks, arXiv preprint arXiv:1511.08458.
Palangi, H., Deng, L., Shen, Y., Gao, J., He, X., Chen, J., Song, X., Ward, R., 2016. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. IEEE/ACM Trans. Audio Speech Lang. Process. 24 (4), 694–707.
Park, H., Kwon, S., Kwon, H.-C., 2010. Complete gini-index text (git) feature-selection algorithm for text classification. In: The 2nd International Conference on Software Engineering and Data Mining. IEEE, pp. 366–371.
Parlak, B., Uysal, A.K., 2023. A novel filter feature selection method for text classification: Extensive feature selector. J. Informat. Sci. 49 (1), 59–78.
Petersen, K., Wohlin, C., Baca, D., 2009. The waterfall model in large-scale development. In: Product-Focused Software Process Improvement: 10th International Conference, PROFES 2009, Oulu, Finland, June 15–17, 2009. Proceedings 10, Springer, pp. 386–400.
Qu, C., Yang, L., Qiu, M., Croft, W.B., Zhang, Y., Iyyer, M., 2019. Bert with history answer embedding for conversational question answering. In: Proceedings of the 42nd international ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1133–1136.
Quinlan, J.R., 1996. Learning decision tree classifiers. ACM Comput. Surv. (CSUR) 28 (1), 71–72.
Rahimi, N., Eassa, F., Elrefaei, L., 2020. An ensemble machine learning technique for functional requirement classification. Symmetry 12 (10), 1601.
Rahimi, N., Eassa, F., Elrefaei, L., 2021. One-and two-phase software requirement classification using ensemble deep learning. Entropy 23 (10), 1264.
Rahman, M.A., Haque, M.A., Tawhid, M.N.A., Siddik, M.S., 2019. Classifying non-functional requirements using rnn variants for quality software development. In: Proceedings of the 3rd ACM SIGSOFT International Workshop on Machine Learning Techniques for Software Quality Evaluation, pp. 25–30.
Ramos, J. et al., 2003. Using tf-idf to determine word relevance in document queries. In: Proceedings of the First Instructional Conference on Machine Learning, vol. 242, Citeseer, pp. 29–48.
Rehman, A., Javed, K., Babri, H.A., 2017. Feature selection based on a normalized difference measure for text classification. Informat. Process. Manage. 53 (2), 473–489.
Rehman, A., Javed, K., Babri, H.A., Asim, M.N., 2018. Selection of the most relevant terms based on a max-min ratio metric for text classification. Expert Syst. Appl. 114, 78–96.
Resnik, P., 1999. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. J. Artificial Intell. Res. 11, 95–130.
Saleem, S., Ghani Khan, M.U., Saba, T., Abunadi, I., Rehman, A., Bahaj, S.A., 2022. Efficient facial recognition authentication using edge and density variant sketch generator. Comput. Mater. Continua 70 (1).
Sayyad Shirabad, J., Menzies, T., 2005. Promise software engineering repository.
Shafiq, M., Tian, Z., Bashir, A.K., Jolfaei, A., Yu, X., 2020. Data mining and machine learning methods for sustainable smart cities traffic classification: A survey. Sustain. Cities Soc. 60, 102177.
Sharma, S., Jain, A., 2023. Hybrid ensemble learning with feature selection for sentiment classification in social media. In: Research Anthology on Applying Social Networking Strategies to Classrooms and Libraries, IGI Global, pp. 1183–1203.
Singh, P., Singh, D., Sharma, A., 2016. Rule-based system for automated classification of non-functional requirements from requirement specifications. In: 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI). IEEE, pp. 620–626.
Tang, S., Yuan, S., Zhu, Y., 2019. Deep learning-based intelligent fault diagnosis methods toward rotating machinery. Ieee Access 8, 9335–9346.
Tiun, S., Mokhtar, U., Bakar, S., Saad, S., 2020. Classification of functional and non-functional requirement in software requirement using word2vec and fast text. Journal of Physics: Conference Series, vol. 1529. IOP Publishing, p. 042077.
Tóth, L., Vidács, L., 2019. Comparative study of the performance of various classifiers in labeling non-functional requirements. Informat. Technol. Control 48 (3), 432–445.
Tutsoy, O., Polat, A., Çolak, Ş., Balikci, K., 2020. Development of a multi-dimensional parametric model with non-pharmacological policies for predicting the covid-19 pandemic casualties. Ieee Access 8, 225272–225283.
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need. Adv. Neural Informat. Process. Syst. 30.
Vlas, R., Robinson, W.N., 2011. A rule-based natural language technique for requirements discovery and classification in open-source software development projects. In: 2011 44th Hawaii International Conference on System Sciences. IEEE, pp. 1–10.
Vlas, R.E., Robinson, W.N., 2012. Two rule-based natural language strategies for requirements discovery and classification in open source software development projects. J. Manage. Informat. Syst. 28 (4), 11–38.
Vogelsang, A., Borg, M., 2019. Requirements engineering for machine learning: Perspectives from data scientists. 2019 IEEE 27th International Requirements Engineering Conference Workshops (REW). IEEE, pp. 245–251.
Wang, X., Jiang, W., Luo, Z., 2016. Combination of convolutional and recurrent neural network for sentiment analysis of short texts. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pp. 2428–2437.
Wang, S., Tang, J., Liu, H., 2017. Feature selection..
Yin, W., Schütze, H., 2016. Multichannel variable-size convolution for sentence classification, arXiv preprint arXiv:1603.04513.
Yin, W., Kann, K., Yu, M., Schütze, H., 2017. Comparative study of cnn and rnn for natural language processing, arXiv preprint arXiv:1702.01923.
Yogatama, D., Dyer, C., Ling, W., Blunsom, P., 2017. Generative and discriminative text classification with recurrent neural networks, arXiv preprint arXiv:1703.01898.
Zanella, L., Facco, P., Bezzo, F., Cimetta, E., 2022. Feature selection and molecular classification of cancer phenotypes: A comparative study. Int. J. Mol. Sci. 23 (16), 9087.
Zhang, Y., Jin, R., Zhou, Z.-H., 2010. Understanding bag-of-words model: a statistical framework. Int. J. Machine Learn. Cybernet. 1, 43–52.
Zhang, Y., Roller, S., Wallace, B., 2016. Mgnc-cnn: A simple approach to exploiting multiple word embeddings for sentence classification, arXiv preprint arXiv:1603.00968.
Zhang, Y., Roller, S., Wallace, B.C., 2016. MGNC-CNN: A simple approach to exploiting multiple word embeddings for sentence classification. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, San Diego, California, pp. 1522–1527. https://doi.org/10.18653/v1/N16-1178. https://aclanthology.org/N16-1178.
Zhou, C., Sun, C., Liu, Z., Lau, F., 2015. A c-lstm neural network for text classification, arXiv preprint arXiv:1511.08630.