Tailoring Random Forest for Requirements Classification

Andreas Falkner(✉), Gottfried Schenner, and Alexander Schörghuber

Siemens AG Österreich, Vienna, Austria
{andreas.a.falkner,gottfried.schenner,alexander.schoerghuber}@siemens.com

Abstract. Automated and semi-automated classifications of requirements (type and topics) are important for making requirements management more efficient. We report how we tailored a random forest approach in the EU funded project OpenReq, aiming for sufficient quality for practical use in bid projects. Evaluation with thirty thousand requirements in English from nine tender documents for rail automation systems in various countries show that user expectations are hard to meet.

Keywords: Random forest · Evaluation · Industrial application · Requirement classification · Text categorization

1 Introduction

Requirements management for large projects is a time-consuming and error-prone task which can be supported by artificial intelligence [10]. In the Horizon 2020 project OpenReq[1], we developed several solution approaches and evaluated them with data from bid projects in the domain of railway safety systems.

Requests for proposal (RFP) or tenders for large infrastructure systems are typically issued by national authorities and comprise natural language documents of several hundred pages with requirements of various kind (domain specific, physical, non-functional, references to standards and regulations, etc.). Preparing a proposal (bid) to answer a tender requires (1) to identify the requirements in the tender text and (2) to assign experts to assess the company's compliance to those requirements. The difficult part is the classification of the (real) requirements w.r.t. predefined topics (which are covered by the experts).

For both tasks, it is important to achieve a very high true positive rate (recall), because requirements which are not detected or are assigned to the wrong experts will not be assessed correctly and may lead to high non-compliance cost. On the other hand, the true negative rate shall also be high, so that unnecessary work is reduced.

The contribution of this work is twofold: Firstly, a new way to tailor the well-known random forest approach [5] by optimizing the model's configuration

[1] http://openreq.eu/.

© Springer Nature Switzerland AG 2020

to requirements classification in general. Secondly, its evaluation in the domain of rail automation (30,000 real-world requirements, 50 topics).

The remainder of this paper is structured as follows: In Sect. 2 we list previous approaches to solve this and similar problems. After presenting our solution in Sect. 3, we report on the evaluation results in Sect. 4. Section 5 summarizes the main outcome and its impact on the users.

2 Related Work

Many approaches to automatic text classification are not specific to requirements management. In the past they tended to be rule-based, but lately (supervised) machine learning has become increasingly popular [13].

Approaches specific to requirements classification vary in the preprocessing NLP pipeline and in their choice of used classifiers. For example, a micro-service for requirements classification developed in the OpenReq project uses Naïve Bayes classifiers [9]. [18] describes a NLP pipeline for extracting requirements from prescriptive documents and uses a SVM classifier to classify the requirements into disciplines.

[14] is an early paper on automatic topic categorization of requirements written in natural language using a bootstrapping approach with machine learning (Naïve Bayes). [17] uses automatic requirement categorisation in an industrial setting to support the review of large natural language specifications in the automotive domain.

Semantic approaches for text classification incorporate not only syntactic but also semantic information, e.g.., provided by systems for automatic information and relation extraction [11]. For a survey of such approaches see [3]. [16] discusses the use of ontologies and semantic technologies in requirements management.

In contrast to new approaches which use pre-trained models, e.g. BERT [8], this work relies solely on traditional machine learning approaches and a model which has been in industrial use for two years.

3 Solution

Text categorization labels paragraphs of natural language documents with predefined categories (or classes). It is a typical application of supervised learning which relies on an initial set of labelled instances used for training [1]. We use binary classification for *type classification* (whether an instance is a requirement or not) and multi-label classification for *topics* (an instance can be assigned to either zero or one or several topics). For example, an input instance to classification is the paragraph "The power supply shall consist of the two sources: one main and one for backup." and the corresponding output could be $requirement = yes$ for binary type classification and $topics = \{Power, Diesel\}$ for multi-label topic classification. Internally, we implemented multi-label classification as multiple isolated binary classification problems [21].

Our solution comprises: (1) a *Random Forest* approach which is a proven classifier for text categorization [1], (2) text preprocessing such as tokenization, n-grams, stop word removal and reduction of word inflections, (3) a feature engineering stage which includes calculating feature weights and selecting relevant features [2], and (4) various sampling strategies to overcome problems with imbalanced data [12].

For tailoring this solution, we evaluated various combinations for these steps and identified the most promising model configuration for application to bid projects (for more details, see [19]):

- As a sampling strategy, we analyzed random under-sampling (RUS) [12], SMOTE [6] and no rebalancing. RUS showed superior performance. For training, we apply RUS ten times with 10 different random seeds, resulting in ten training sets. One model is trained per training set and all ten models are finally aggregated using majority vote.
- To remove word inflection, lemmatization (StanfordNLP [15]) [20] and stemming (Porter Stemmer) [20] were compared. Although evaluation revealed that using the lemmatizer increases performance, we decided on the stemmer because of its less restrictive software licence.
- We evaluated usage of tokens based on n-grams, $n \in \{1\}$, $n \in \{1, 2\}$, .., $n \in \{1, 2, 3, 4, 5\}$. This parameter has no significant influence on performance – therefore uni-grams are used to keep feature space small.
- Different feature weights were compared: set of words [21], term frequency (TF) [20], TF-IDF [20] and (R)TF-IGM [7]. As this parameter did not show significant influence on performance, we selected TF because of its algorithmic simplicity.
- Using a stop-word list [20] from the Natural Language Toolkit[2] increased performance.
- The following common feature selection methods were evaluated: information gain (IG), χ^2 and term frequency (TF) [21]. TF showed good results and a fast runtime. The algorithm is configured to keep 1,300 features.

Additionally to the described configurations above, a user can set a threshold for positive classification before starting the predictor. Only if the built model's probability for a requirement is greater than or equal to the given threshold, the requirement is classified as positive instance. By that, the priority of true positives versus true negatives can be decided [23].

4 Evaluation

After having tailored the random forest approach including text preprocessing, we evaluated it using previous bid projects provided by the bid group. In addition to a quantitative evaluation, we did a small field study with three experts (unstructured interviews, application to a new, yet unlabelled bid project).

[2] http://www.nltk.org/, accessed 09.01.2020.

4.1 Data Set

The data set used for evaluation comprises the text paragraphs of nine tender documents. All of them were written in English (most of them translated from a native language). Each entry was labelled by experts as a requirement (or non-requirement) and assigned to relevant topics (mostly between one and three, out of 52 potential topics). Thereafter, we randomly chose six documents as training data, resulting in a 47% test data split. Table 1 lists the numbers of requirements, non-requirements, and assigned topics for training data as a whole and for test data separately for each project[3] and in total.

Concerning type classification, 14,714 out of 17,556 potential requirements are labeled as requirement, leading to a prevalence of 84%.

From 52 potential topics, 50 occur in the training data, and 34 occur in the test data. Depending on prevalence in the training data, we selected three groups of 5 topics each: A (5/6) comprises all topics which occur more than 1,000 times in the training data (and at least once in the test data). For B (5/17), we chose – from all topics which occur more than 200 times in the training data – those which occur at least 500 times in the test data or in all three test projects. For

Table 1. Test data – numbers of types and topics

		Training data	Test data	Project 1	Project 2	Project 3
Total		17,556	8,256	978	6,465	813
Req		14,714	6,923	867	5,336	720
Non-Req		2,842	1,333	111	1,129	93
52 topics		20,288	10,479	867	8,892	720
PM	A	4,710	1,663	683	432	548
IXL		2,073	338	0	338	0
MMI		1,590	1,287	0	1,287	0
Power		1,448	405	47	299	59
BidManager		1,141	56	0	56	0
LED	B	507	361	116	132	113
SCADA		476	1,317	0	1,317	0
Diagnosis		422	820	0	820	0
SystemMgmt		400	548	0	548	0
Engineering		203	1,328	0	1,328	0
GSMR	C	172	181	13	168	0
Commissioning		39	70	0	70	0
PIS		27	223	0	223	0
Diesel		8	135	0	135	0
EHS		8	195	0	195	0

[3] Project names are confidential – therefore we use numbers 1 to 3.

C (5/27), we selected – from all topics which occur less than 200 times in the training data – those which occur most often in the test data.

4.2 Metrics

We use (standard) metrics which help to directly judge user benefit: recall (sensitivity, true positive rate, TPR), specificity (true negative rate, TNR) [22], receiver operating characteristics (ROC) curve analysis [4] and custom metrics for estimating time savings. For bid projects, a high recall is very important in order to reduce risk of high non-compliance cost due to ignorance of information. Specificity, on the other hand, is important to avoid unnecessary work due to wrongly assigned topics. All metrics are micro-averaged, i.e., summing all quantities and then calculating the metrics on the sum. This leads to a combined metric for all test data and a combined metric for multi-labels per topic.

For estimating the time savings, we compare our solution to the decisions by a requirements manager, using the metrics defined by Eqs. 1 and 2, which are based on the time to comprehend a requirement ($t_{analyze}$), the time to change a label (t_{change}) and standard evaluation quantities: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). Assuming that an expert does not make any mistakes, he or she needs to analyze each requirement and set only the positive labels. In the automated approach, true positives need not be analyzed nor set, but additional work is necessary: For type classification (td_{type}), true and false negatives are still analyzed by the requirements manager and false negatives are set to positive in order to get TPR high – this has no effect in Eq. 1. False positives must be changed to negative by topic experts. For topic classification (td_{topic}), false positives and negatives are corrected by the topic experts during assessment. If the values for $t_{analyze}$ and t_{change} are known (e.g., as seconds per requirement on average) then the difference in hours can be calculated.

$$td_{type} = (FP - TP) \times t_{change} - (TP + FP) \times t_{analyze} \tag{1}$$

$$td_{topic} = FP \times t_{analyze} + (FP - TP) \times t_{change} \tag{2}$$

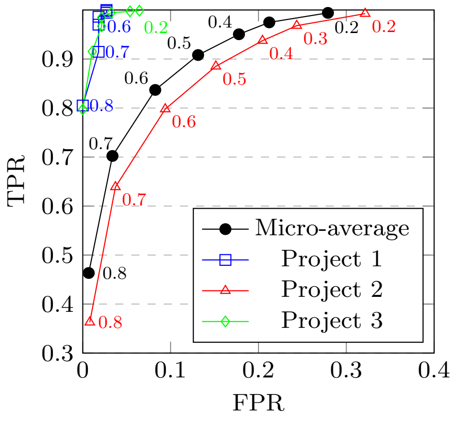
4.3 Type Classification

We used various thresholds (20%, 30%, ..., 80%) to get a feeling of the balance of TPR and TNR – see the ROC curve in Fig. 1a. Projects 1 and 3 perform very well, probably because they have similar properties as projects in the training set (same author, different stations on the same railway line).

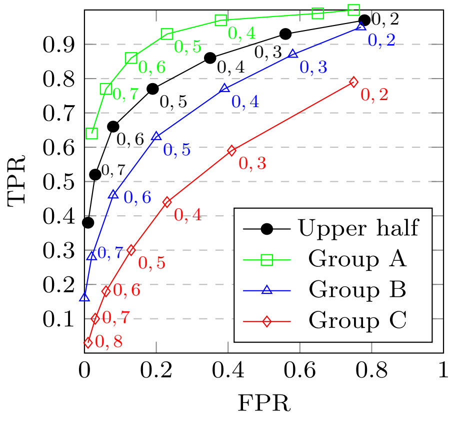
In our interviews, the requirements managers turned out to be very risk-averse. Therefore, they prefer a very high TPR (e.g., 99%, as achieved with threshold 20%) and accept the relatively low TNR of 72% compared to threshold 50% (with fairly balanced TPR of 91% and TNR of 87%) – see Table 2. Assuming $t_{analyse} = 30$ s and $t_{change} = 5$ s, the time savings are more than 60 working hours for the three test projects. This equates to savings of approximately one working day for each thousand requirements which was confirmed in a field study with a new bid project.

Table 2. Evaluation results – TPR, TNR and estimated time savings

	Threshold 50%				Threshold 20%			
	Micro-avg	P1	P2	P3	Micro-avg	P1	P2	P3
TPR (%)	91	99	89	98	99	100	99	100
TNR (%)	87	98	85	98	72	97	68	94
td_{type} (h)	−62	−8	−47	−7	−69	−8	−54	−7



(a) type classification



(b) topic classification

Fig. 1. ROC curves

4.4 Topic Classification

Prevalence of topics is very low – the average in the training data was lower than 2% and even for the 5 most common topics only around 6%. The ROC curve in Fig. 1b shows that prediction quality on average (for the upper half of topics) is not as good as for type classification. The groups B and C from Table 1 perform much worse than group A (with comparably higher prevalence).

In our interviews, the requirements managers preferred a threshold of 50% - see Table 3. However, they judged the achieved TPR of 73% (micro-average of all topics occurring in at least one test project) as too low for practical use. Even the TPR of 77% (and TNR 81%) for the upper half of topics was not sufficient, as nearly 2,100 topic assignments are missing and more than 16,300 assignments are wrong. Again, projects 1 and 3 perform much better than the more typical project 2. The new project from the field study performed similar to the latter.

Although the requirements managers do not need to spend any time for topic assignment, the metric td_{topic} estimates an additional effort of 127 h for the necessary manual adjustments by topic experts. Only for project 3 or with a high threshold can time savings be achieved.

Table 3. Evaluation results – TPR and TNR at threshold 50%

	TPR				TNR			
	Micro-avg	P1	P2	P3	Micro-avg	P1	P2	P3
34 topics in test	73	98	69	100	81	92	81	98
Upper half	77	99	73	100	81	97	80	98
5 topics A	93	100	89	100	77	96	75	97
5 topics B	63	99	61	100	80	99	78	99
5 topics C	30	38	30		87	76	87	

5 Conclusion

We chose the random forest approach for requirements classification because it is easier to maintain and deploy than advanced deep learning solutions. Training is less expensive and can be done on local servers.

The results were fairly good for type classification and topics with a prevalence $> 5\%$ (better than, e.g.., an alternative approach based on Naïve Bayes). Application in a field study showed a high potential for reducing efforts for requirements managers (e.g.., 80% of the time for type classification). However, improvements – especially for topics with a low prevalence $< 3\%$ – are necessary to fulfil the users' demand for high TPR and TNR, i.e., both $>> 95\%$.

Acknowledgments. The work presented here has been conducted in the scope of the Horizon 2020 project OpenReq, supported by the European Union under Grant Nr. 732463.

References

1. Aggarwal, C.C.: Machine Learning for Text. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-73531-3
2. Allahyari, M., et al.: A brief survey of text mining: classification, clustering and extraction techniques. CoRR abs/1707.02919 (2017)
3. Altınel, B., Ganiz, M.C.: Semantic text classification: a survey of past and recent advances. Inf. Process. Manag. **54**(6), 1129–1153 (2018)
4. Bekkar, M., Djemaa, H.K., Alitouche, T.A.: Evaluation measures for models assessment over imbalanced datasets. J. Inf. Eng. Appl. **3**(10), 27–38 (2013)
5. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
6. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: synthetic minority over-sampling technique. J. Artif. Intell. Res. **16**, 321–357 (2002)
7. Chen, K., Zhang, Z., Long, J., Zhang, H.: Turning from TF-IDF to TF-IGM for term weighting in text classification. Expert Syst. Appl. **66**, 245–260 (2016)
8. 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)

9. Falkner, A., Palomares, C., Franch, X., Schenner, G., Aznar, P., Schoerghuber, A.: Identifying requirements in requests for proposal: a research preview. In: Knauss, E., Goedicke, M. (eds.) REFSQ 2019. LNCS, vol. 11412, pp. 176–182. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-15538-4_13
10. Fucci, D., et al.: Needs and challenges for a platform to support large-scale requirements engineering. A multiple case study. CoRR abs/1808.02284 (2018)
11. Gupta, P., Schütze, H., Andrassy, B.: Table filling multi-task recurrent neural network for joint entity and relation extraction. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pp. 2537–2547 (2016)
12. Haixiang, G., Yijing, L., Shang, J., Mingyun, G., Yuanyue, H., Bing, G.: Learning from class-imbalanced data: review of methods and applications. Expert Syst. Appl. **73**, 220–239 (2017)
13. Kadhim, A.I.: Survey on supervised machine learning techniques for automatic text classification. Artif. Intell. Rev. **52**(1), 273–292 (2019)
14. Ko, Y., Park, S., Seo, J., Choi, S.: Using classification techniques for informal requirements in the requirements analysis-supporting system. Inf. Softw. Technol. **49**(11–12), 1128–1140 (2007)
15. Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S.J., McClosky, D.: The Stanford CoreNLP natural language processing toolkit. In: Association for Computational Linguistics (ACL) System Demonstrations, pp. 55–60 (2014)
16. Moser, T., Winkler, D., Heindl, M., Biffl, S.: Requirements management with semantic technology: an empirical study on automated requirements categorization and conflict analysis. In: Mouratidis, H., Rolland, C. (eds.) CAiSE 2011. LNCS, vol. 6741, pp. 3–17. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21640-4_3
17. Ott, D.: Automatic requirement categorization of large natural language specifications at Mercedes-Benz for review improvements. In: Doerr, J., Opdahl, A.L. (eds.) REFSQ 2013. LNCS, vol. 7830, pp. 50–64. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37422-7_4
18. Pinquié, R., Véron, P., Segonds, F., Croué, N.: Requirement mining for model-based product design. Int. J. Product Lifecycle Manag. **9**(4), 305–332 (2016)
19. Schörghuber, A.: Classification of requirements in the tender process. Master's thesis, University of Technology Vienna, Vienna (2019)
20. Schütze, H., Manning, C.D., Raghavan, P.: Introduction to Information Retrieval, vol. 39. Cambridge University Press, Cambridge (2008)
21. Sebastiani, F.: Machine learning in automated text categorization. ACM Comput. Surv. (CSUR) **34**(1), 1–47 (2002)
22. Sokolova, M., Lapalme, G.: A systematic analysis of performance measures for classification tasks. Inf. Process. Manag. **45**(4), 427–437 (2009)
23. Tosun, A., Bener, A.: Reducing false alarms in software defect prediction by decision threshold optimization. In: Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement, pp. 477–480 (2009)