Towards Just-In-Time Feature Request Approval Prediction

Feifei Niu, Chuanyi Li, Heng Chen, Jidong Ge, Bin Luo

State Key Laboratory for Novel Software Technology,

Software Institute, Nanjing University, China

Background-JITFRAP

Feature request approval prediction (FRAP): predict whether a feature request will be accepted or rejected by the software manager

Just-in-time FRAP (JITFRAP): predict as soon as it is proposed

Related Work

Nizamani et al. defined the problem as the machine learning based binary classification problem.

$$c = f(r); c \in \{approve, reject\}, r \in R$$

Ramírez-Mora et al. predicted whether an issue would success or not with the help of comments under each issue.

Characteristics

Timing Affected

Just-in-time

Affected by project status

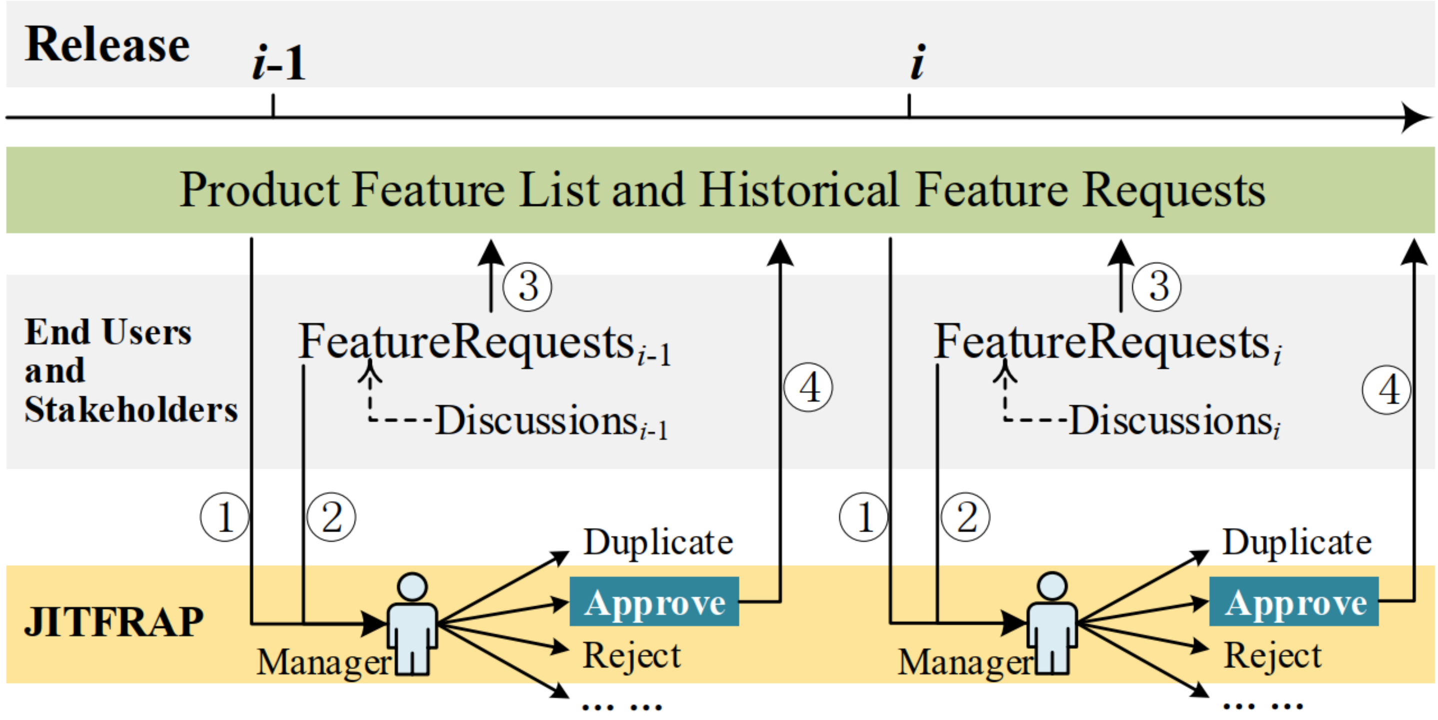

Fig. 1. Scenario of feature request approval prediction.

南京大学 NANJING UNIVERSITY

Data Set Generation

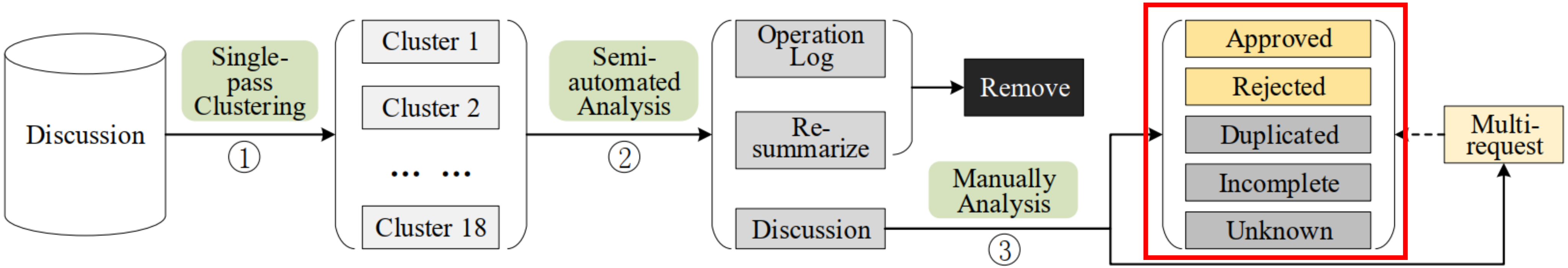

Fig. 2. Pipeline for deriving types of approval decisions from discussions.

Table 1. # of different feature requests in the data set.

Types	After Round 1	After Round 2	Final
Approved	13306	13840	***10727***
Rejected	4829	5224	***4127***
Duplicated	1884	1906	–
Incomplete	577	616	–
Unknown	8166	8732	–
Multi-Request	478	–	–

Available at https://zenodo.org/record/6544368

Impact of Omitting Timing Related

Timing-related refers to that we should never train a classifier using the post-proposed feature requests to predict pre-proposed ones.

A1: It would be nice if we could have multiple auto-type entries. ***Approved***

B1: Consider a way to sort the Auto-Type Entry Selection window's content. This would be useful when multiple entries exist for the current window. ***Approved***

A2: I think it would be nice to be able to type in a password to get in, if your key (floppy/usb/cd/etc.) is lost or unavailable. ***Approved***

B2: One feature that would be incredibly helpful for using KeePass in a corporate environment would be the ability to have more than one master password that can open the same database, or the ability to open the same database either with a password or with a key file. ***Rejected***

Goal: The existence of timing would influence the results.

Impact of Omitting Timing Related

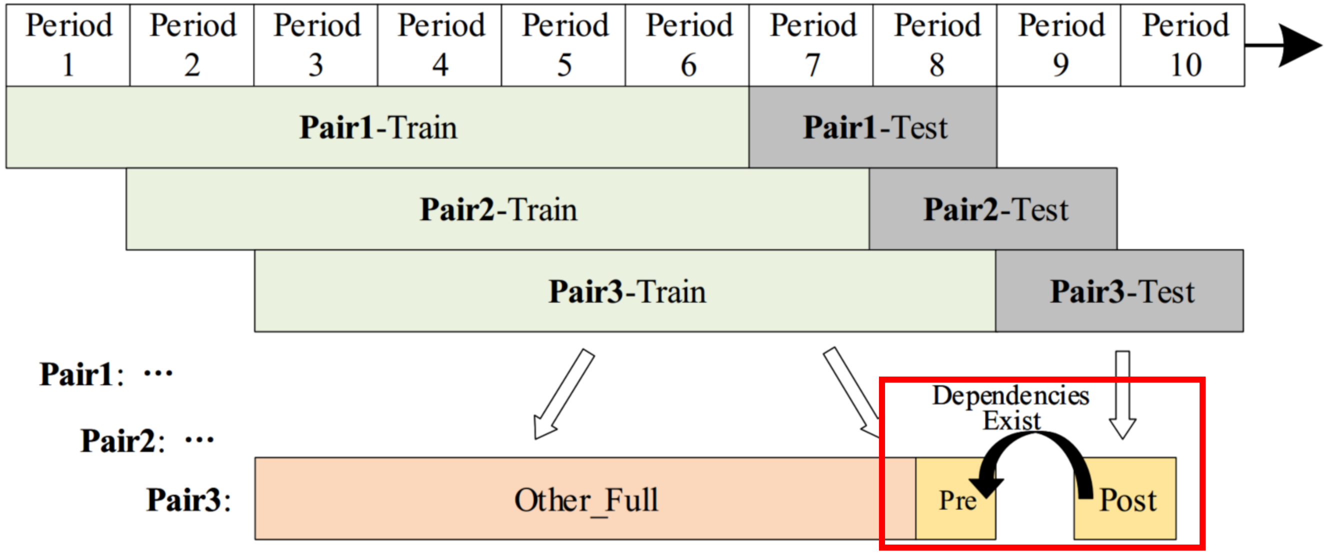

Fig. 5. Methods of preparing experimental data.

Table 2. Comparisons of omitting timing.

Train*	Test	Approved (%) Precision	Approved (%) Recall	Rejected (%) Precision	Rejected (%) Recall
Other_Full	Post	74.4	98.2	71.0	11.4
Other_P1+Pre	Post	74.1 ↓	98.4 ↑	69.6 ↓	9.9 ↓
Other_Full	Pre	76.3	97.6	51.4	7.9
Other_P2+Post	Pre	77.2 ↑	96.6 ↓	55.9 ↑	13.1 ↑

* For all training sets, the sizes and the ratios of Approved and Rejected data instances are the same.

NANJING UNIVERSITY

Impact of Omitting Just-In-Time

Just-in-time means that upon the request is made, the prediction should be determined.

Goal: Taking comments/discussions into consideration would derive the different performance of the classifier.

Data Set	Inputut	Approved(%)			Rejected(%)		
		Precision	Recall	F1	Precision	Recall	F1
Our data set	Text+All_Discussion	79.0	90.55	84.4	69.51	47.2	56.2
	Text+User_Discussion	70.1	98.92	82.3	76.09	7.5	13.7
	Text	70.4	98.78	82.2	76.85	8.9	16.0

Data Set	Input	Success(%)			Fail(%)		
		Precision	Recall	F1	Precision	Recall	F1
In [10]	Text+All_Discussion	25.1	0.24	10.7	87.59	97.1	92.4
	Text	16.7	6.7	0.5	88.03	99.8	93.3

Impact of Omitting Project Status

By status, we mean the background knowledge of the project, such as introduction, feature list, changelog, etc.

Goal: Taking the status of the project into consideration would improve the prediction performance.

Input	Approved			Rejected		
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**
Text	70.4	98.78	82.2	76.85	8.9	16.0
Changelog_All	67.9	47.5	54.3	73.6	63.5	66.5
Text+Changelog_All	74.5	89.8	81.4	59.5	32.8	42.3
Text+Changelog_Simi	72.4	97.2	83.0	75.2	18.8	30.1

Baseline Approaches

RQ1: What are the different performances using different inputs, different feature extraction methods, and different classification algorithms?

RQ2: To what extent can automated approaches identify approved feature requests from rejected ones?

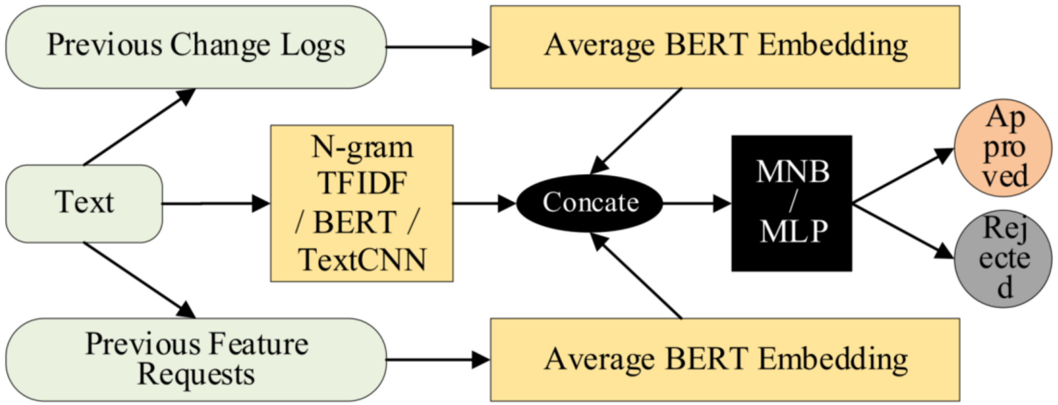

Fig. 6. Framework of the proposed basic approaches.

I. TFIDF(Text) + BERT(CL) + BERT(SimiFR) ⟶ MNB,
II. BERT(Text) + BERT(CL) + BERT(SimiFR) ⟶ MLP,
III. TextCNN(Text) + BERT(CL) + BERT(SimiFR) ⟶ MLP.

Experimental Results

Table 4. Performances of the proposed basic approaches.

Approach	Input	Feature Extraction	Algorithm	Approved (%)			Rejected (%)		
				Precision	Recall	F1	Precision	Recall	F1
I	Text	TFIDF	MNB	70.4	**98.8**	82.2	**76.9**	8.9	16.0
	Text+CL	TFIDF		73.0	96.7	83.2	74.9	21.7	33.7
	Full (Text+CL+SimiFR)	+ BERT		73.2	96.7	**83.3**	75.6	22.6	34.8
II	Text	BERT	MLP + softmax	72.0	93.9	81.5	60.0	20.0	30.0
	Text+CL			73.3	94.4	82.5	67.0	24.7	36.1
	Full			73.2	95.3	82.8	69.5	23.5	35.1
III	Text	TextCNN	MLP + softmax	74.2	88.9	80.9	57.4	32.6	41.5
	Text+CL	TextCNN + BERT		75.4	89.7	81.9	61.5	36.1	45.5
	Full			76.0	90.1	82.5	63.6	37.8	47.4
	Full w/o Simi_A			**79.1**	79.4	79.3	54.6	**54.2**	**54.4**
	Full w/o Simi_R			78.2	80.5	79.3	54.5	51.1	52.7

Findings:

Combination of text, changelog and similar feature requests can enhance the classifier.

The best result was achieved by MLP+Full w/o Simi_A+TextCNN

Conclusion and Future Work

Conclusion:

(1) We contribute a standard data set.

(2) We conduct an empirical analysis with pre-experiments.

(3) We present and discuss preliminary results of three basic approaches.

Future Work:

(1) We will integrate duplicate/incomplete detection.

(2) We will work to improve and expand our data set.

THANKS

Q&A

Feifei Niu, Chuanyi Li, Heng Chen, Jidong Ge, Bin Luo