USING DOMAIN-SPECIFIC WORD EMBEDDINGS TO BOOST KEYWORD-BASED COMMIT CLASSIFICATION

Tjaša Heričko and Boštjan Šumak
Faculty of Electrical Engineering and Computer Science, University of Maribor

BACKGROUND

During a software's lifetime, source code management tools facilitate managing software changes. Changes (i.e., commits) are performed for various purposes [1] and are accompanied by short messages from committers communicating the code changes in natural text:

- **adaptive** commit (to adapt to changes in the environment)
  Example message: "Implement Scheduler method with dueTime"
- **corrective** commit (to fix bugs, faults, and defects)
  Example message: "Fix autoConnect calling onStart twice"
- **perfective** commit (to improve software quality attributes)
  Example message: "Refactor test to use CountDownLatch instead of Thread.sleep"

CHALLENGE

Keywords extracted from commit messages are good indicators of commit intent. The existing keyword-based approach uses a set of 20 keywords, defined based on word frequency analysis, to classify commits [2]. However, developers often use jargon terms, acronyms, misspelled words, and synonyms or related words, making it challenging to classify based on a small set of fixed keywords alone.

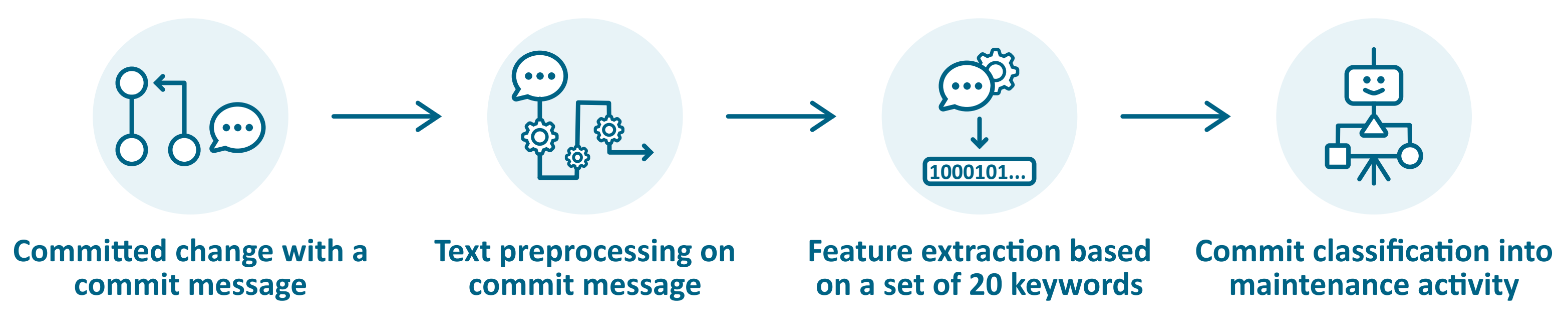
Committed change with a commit message
Text preprocessing on commit message
1000101...
Feature extraction based on a set of 20 keywords
Commit classification into maintenance activity

Figure 1. *A high-level overview of the existing keyword-based approach*

PROPOSED APPROACH

We propose an enhancement to the keyword-based commit classification approach by extending the set of keywords by exploiting semantic similarities between the words of commit messages. Word embedding vectors were generated by training on a domain-specific corpus containing 2.5 million commit messages from 500 code repositories [3].

Algorithm 1. *The feature extraction process of the proposed approach*

```
Input: list of keywords: $keywords = [k_1, k_2, ..., k_{20}]$; limit of the most similar words to target keyword: $N$; adjusted limit of the most similar words to target keyword without duplications: $M$; domain-specific word embedding model: $embedding\_model$; commit message: $message$; labeled dataset: $dataset$; similarity threshold: $t$
Output: extracted features $features(N,M,t) = [feature_{k_1}, feature_{k_2}, ..., feature_{k_{20}}]$
for each keyword $k$ in $keywords$ do
    $extended\_keyword_{k,N} = [[w_{k,1}, sim(k,w_{k,1})], ..., [w_{k,N}, sim(k,w_{k,N})]]]) \leftarrow$ for target keyword $k$ find $N$ most similar words $w_{k,i\in\{1,...,N\}}$ in $embedding\_model$ based on cosine similarity $sim(k,w_{k,i})$
for each $k \in \{1, ..., 20\}$ in $extended\_keywords_{k,N}$ do
    $m \leftarrow 0$
    while $m < M$ do
        for each word $w_{k,i}$ in $[w_{k,i}, sim(k,w_{k,i})]$ do
            if $w_{k,i}$ is equal to $k'$ then
                continue with next word
            if $w_{k,i}$ is equal to $w'_{k',i'}$ then
                if $sim(k,w_{k,i}) < sim(k',w'_{k',i'})$ then
                    continue with next word
            append $[w_{k,i}, sim(k,w_{k,i})]$ to $extended\_keyword_{k,M}$
            $m \leftarrow m + 1$
for each $message$ in $dataset$ do
    for each $k \in \{1, ..., 20\}$ in $extended\_keyword_{k,M}$ do
        if $message$ contains keyword $k$ or any word $w_{k,i}$ where $sim(k,w_{k,i}) >= t$ then
            $feature_k \leftarrow$ True
        else
            $feature_k \leftarrow$ False
```

Table 1. *Examples of extended keywords (N=15, M=5)*

Keyword	The *M* most similar words to target keyword
add	ad *(sim=0.93)*, updat *(sim=0.63)*, extend *(sim=0.61)*, introduc *(sim=0.60)*, includ *(sim=0.55)*
npe	nullpointerexcept *(sim=0.89)*, concurrentmodificationexcept *(sim=0.73)*, classcastexcept *(sim=0.72)*, crash *(sim=0.71)*, indexoutofboundsexcept *(sim=0.68)*
refactor	simplifi *(sim=0.76)*, extract *(sim=0.71)*, cleanup *(sim=0.67)*, move *(sim=0.65)*, factor *(sim=0.65)*

PRELIMINARY RESULTS

While the existing approach *(t=1)* showed that only 65% of the sample commit messages examined contained at least one keyword, the applied approach demonstrated that up to 82% contained at least one of the extended keywords.

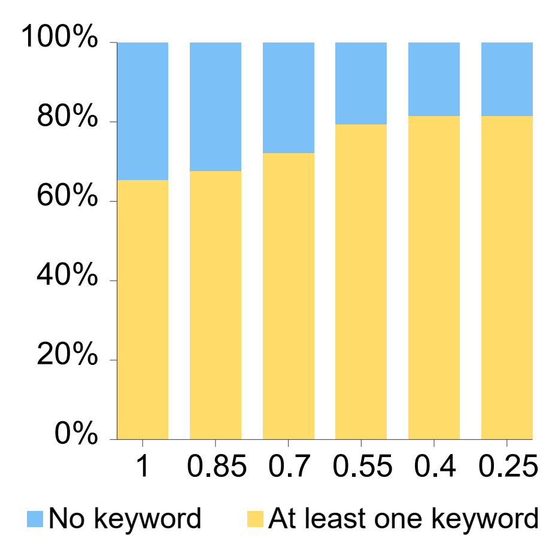

Figure 2. *Keywords presence in sample commits considering different similarity thresholds (N=15, M=5)*

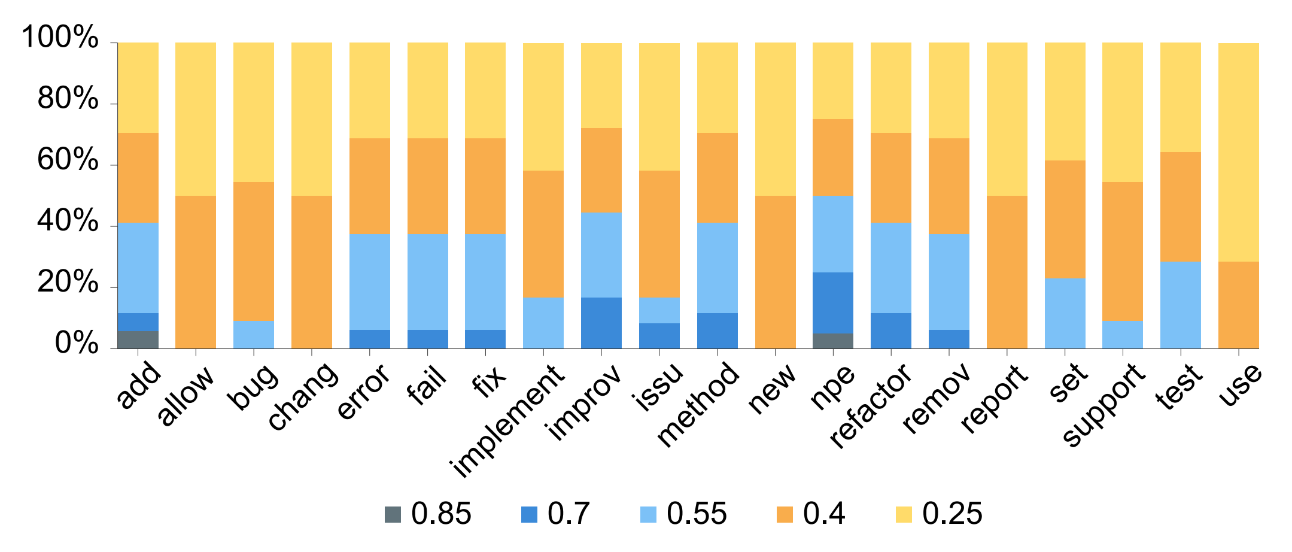

Figure 3. *Per-keyword distribution of the number of included words extending keywords considering different similarity thresholds (N=15, M=5)*

To evaluate the proposed approach, different machine learning models were built on a labeled dataset of 1793 commits [4] using Random Forest, Gradient Boosting Machine, and CART algorithms.

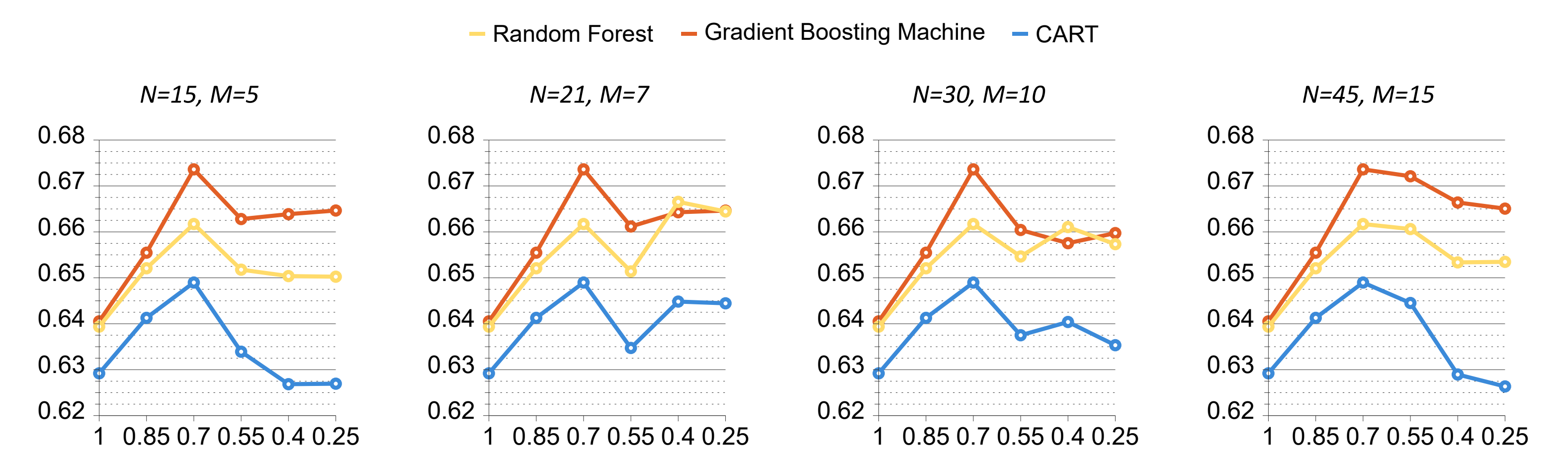

Figure 4. *The performance of classification models (mean F-measure) with 3-times repeated 5-fold cross-validation*

Comparison with the existing approach showed that enriching keywords with the words semantically closest to them can benefit the predictive performance of the models while providing deeper insight into how developers use written communication through commit messages.

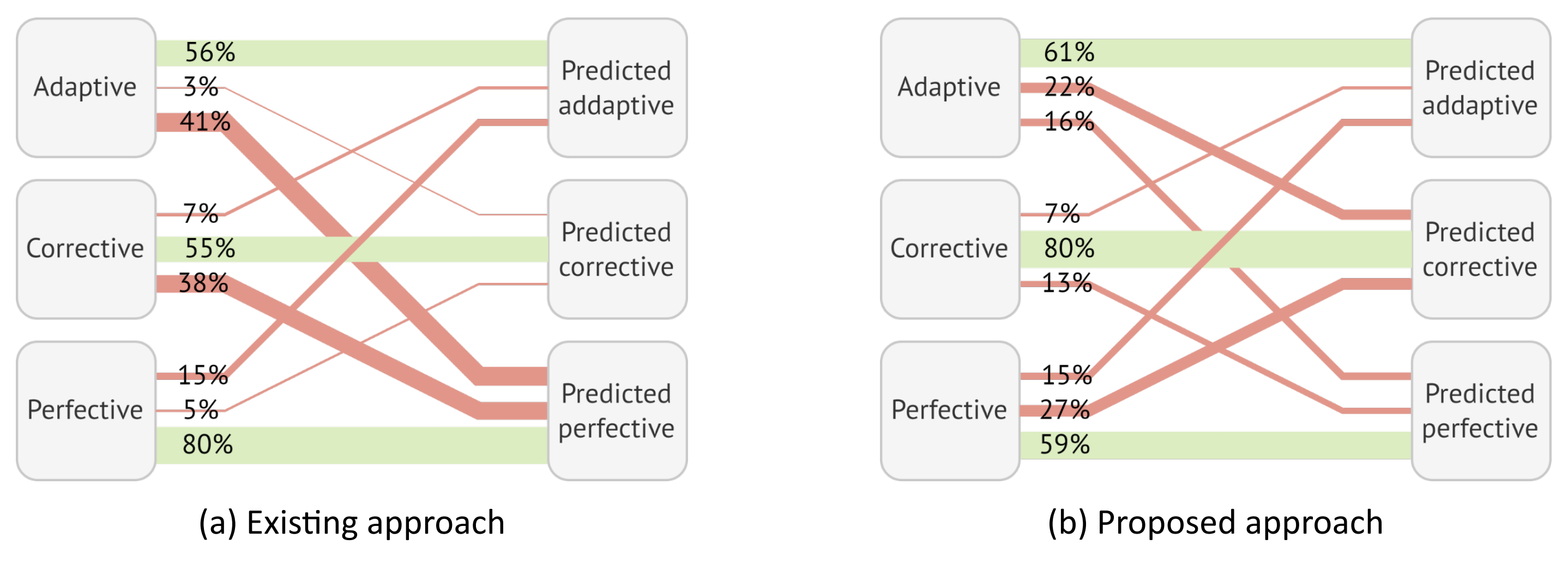

Figure 5. *A comparison of per-class accuracy of the classification models using Gradient Boosting Machine between the existing (a) and the proposed (b) approach (N=15, M=5, t=0.55)*

ACKNOWLEDGMENTS

The authors acknowledge the financial support from the Slovenian Research Agency (Research Core Funding No. P2-0057).

REFERENCES

[1] E. B. Swanson, "The dimensions of maintenance," in Proceedings of the 2nd International Conference on Software Engineering, ser. ICSE'76, 1976.

[2] S. Levin and A. Yehudai, "Boosting automatic commit classification into maintenance activities by utilizing source code changes", in *Proceedings of the 13th International Conference on Predictive Models and Data Analytics in Software Engineering*, ser. PROMISE, 2017.

[3] T. Heričko, S. Brdnik, and B. Šumak, "Commit classification into maintenance activities using aggregated semantic word embeddings of software change messages," in *Proceedings of the Ninth Workshop on Software Quality Analysis, Monitoring, Improvement, and Applications*, ser. CEUR Workshop Proceedings, 2022.

[4] L. Ghadhab, I. Jenhani, M. W. Mkaouer, and M. Ben Messaoud, "Augmenting commit classification by using fine-grained source code changes and a pre-trained deep neural language model," *Information and Software Technology*, 2021.

A: Koroška cesta 46, 2000 Maribor, Slovenia **E:** tjasa.hericko@um.si **T:** +386 (2) 22 07 298