The Journal of Systems & Software 195 (2023) 111513

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

TitleGen-FL: Quality prediction-based filter for automated issue title generation[☆]

Hao Lin[a], Xiang Chen[a,d,*], Xuejiao Chen[a], Zhanqi Cui[b], Yun Miao[a], Shan Zhou[c], Jianmin Wang[c], Zhan Su[a]

[a] School of Information Science and Technology, Nantong University, Nantong, China
[b] Computer School, Beijing Information Science and Technology University, Beijing, China
[c] Technology and Engineering Center for Space Utilization, Chinese Academy of Sciences, Beijing, China
[d] Key Laboratory of Safety-Critical Software (Nanjing University of Aeronautics and Astronautics), Ministry of Industry and Information Technology, Nanjing, China

ARTICLE INFO

Article history:
Received 27 April 2022
Received in revised form 31 July 2022
Accepted 12 September 2022
Available online 17 September 2022

Keywords:
Issue title generation
Quality prediction
Filter
Deep learning
Information retrieval

ABSTRACT

To automatically generate the issue title, researchers formulated the problem into a one-sentence summarization problem and then proposed an effective method iTAPE. However, after analyzing the quality of the titles generated by the method iTAPE, which is measured by ROUGE-L, we find the ROUGE-L scores of only 42.7% of titles can exceed 0.3. This means the quality of the generated titles is not satisfactory and can limit the practicability of the method iTAPE. Therefore, we propose a quality prediction-based filter TitleGen-FL, which can predict whether the method iTAPE can generate a high-quality title after analyzing the issue body. To achieve this goal, TitleGen-FL includes the deep learning-based (DL) module and the information retrieval-based (IR) module. If either of these two modules predict that the high-quality title cannot be generated by iTAPE, TitleGen-FL can automatically filter this issue and return a warning message. To evaluate the effectiveness of our proposed filter, we select the benchmark dataset gathered from real-world open-source projects as our experimental subject. Both automatic evaluation and human study show that our proposed filter TitleGen-FL can effectively filter the issues, which cannot generate high-quality titles by iTAPE.

1. Introduction

A bug report is a specific report, which contains failure description information, stack traces, and other diagnostic information to help developers localize and fix bugs in the software project. To improve the performance of the bug report analysis, developers should follow the guidelines to improve the bug report quality (such as the guidelines provided by Mozilla[1]). But the effect of this treatment is still limited. Therefore, bug report management is an important task during software project development and maintenance (Zhang et al., 2016, 2015). Until now, bug report analysis (such as bug triage (Anvik and Murphy, 2011; Xuan et al., 2014), duplication bug report detection (Nguyen et al., 2012; Alipour et al., 2013), information retrieval-based bug localization (Zhou et al., 2012; Mills et al., 2018, 2020; Saha et al., 2013)) has attracted the wide attention of researchers.

In this study, we focus on the issue quality problem in GitHub.[2] To improve the issue quality, we mainly investigate the automated generation of the issue title, which is the compulsory and important field of the issue. Based on the suggestions from the professional QA platform Testlio, a high-quality issue title should provide a concise and precise summarization of the issue body. Then developers can quickly understand the core content of the bug without reading the details of the issue body. However, the quality of the issue titles in open-source projects is still far from satisfactory due to the limited project budget or the lack of developer experience.

To help developers generate high-quality issue titles, Chen et al. (2020) were the first to formulate the issue title generation problem into a one-sentence summarization problem. Then they proposed the method iTAPE based on a seq2seq model (Sutskever

☆ Editor: Prof Neil Ernst.
* Corresponding author at: School of Information Science and Technology, Nantong University, Nantong, China.

E-mail addresses: linhao56@126.com (H. Lin), xchencs@ntu.edu.cn (X. Chen), 2015110322@stmail.ntu.edu.cn (X. Chen), czq@bistu.edu.cn (Z. Cui), miaoyun001my@gmail.com (Y. Miao), zhoushan@csu.ac.cn (S. Zhou), wangjm@csu.ac.cn (J. Wang), N00Borzs@gmail.com (Z. Su).

1 https://bugzilla.mozilla.org/page.cgi?id=bug-writing.html

2 Notice the issue in GitHub plays the same role as the bug report in bug tracking systems (such as Bugzilla, JIRA, Mantis).

https://doi.org/10.1016/j.jss.2022.111513
0164-1212/© 2022 Elsevier Inc. All rights reserved.

et al., 2014). Moreover, to alleviate the low term frequency problem caused by the identifiers and version numbers in the issue body, they adopted the copy mechanism (Gu et al., 2016; See et al., 2017) and proposed the human-named token tagging method. To show the effectiveness of their proposed method, they gathered issues from the repositories with the top-200 number of stars and constructed a benchmark dataset based on three heuristic rules, which could guarantee the quality of their gathered issues.

In their empirical study, the method iTAPE (Chen et al., 2020) could achieve promising performance. However, after analyzing the quality of the titles generated by the method iTAPE in terms of ROUGE-L (Lin, 2004) (ROUGE-L considers sentence-level structure similarity and automatically identifies the longest co-occurring in the sequence *n*-grams and the detailed introduction can be found in Section 4.2), we found that only 42.7% of the generated titles can exceed 0.3 in terms of ROUGE-L. This means a certain proportion of low-quality titles were generated by the method iTAPE, which may mislead developers, require many efforts to confirm the correctness of these titles, and finally reduce developers' confidence in the method iTAPE. To improve the quality of the issue title, one possible solution is to improve the performance of the method iTAPE (such as improving the quality of the dataset for model training or designing new issue title generation methods by considering more advanced deep learning models). However, we aim to solve this problem from another perspective. In particular, we propose a quality prediction-based filter `TitleGen-FL` to make the method iTAPE more practical. `TitleGen-FL` aims to effectively filter the issues, which may not generate high-quality titles. To perform quality prediction, `TitleGen-FL` contains two modules: the deep learning-based (DL) module and the information retrieval-based (IR) module. After we analyze the main reasons for the low ROUGE-L score in a manual way, we find it is difficult to enumerate all patterns. Therefore, we design the DL module, which can automatically learn the features from a large-scale training set after analyzing the quality of the titles generated by iTAPE. Moreover, if there are no historical issues that are sufficiently similar to the new issue, it is also difficult for the method iTAPE to generate a high-quality title for this new issue. Therefore, we design the IR module, which can compute the IR score based on the similarity between the new issue body and the bodies in historical issues. Given the new issue body, these two modules can both estimate the probability of generating a high-quality title. If either module predicts that the high-quality title may not be generated, `TitleGen-FL` can automatically filter this issue and give a warning message, otherwise, we return the issue title generated by the method iTAPE.

To evaluate the effectiveness of our proposed filter, we select the benchmark dataset shared by Chen et al. (2020) as our experimental subject. This dataset was constructed by gathering bug reports from real-world open-source projects on GitHub. Based on the automatic evaluation way, we find our proposed filter `TitleGen-FL` can effectively filter 56.59% of issues. Notice the high filter ratio is caused by the characteristics of our used experimental subject. For the preserved issues, `TitleGen-FL` improves the performance by 10.87%, 19.47%, and 11.22% in terms of ROUGE-1, ROUGE-2, and ROUGE-L. Then we analyze the issues preserved by `TitleGen-FL` with only the DL module or `TitleGen-FL` with only the IR module, we find these two modules have a certain diversity in the preserved issues. Later, compared to `TitleGen-FL` with only the DL module, `TitleGen-FL` can improve its performance by 2.31%, 5.61%, and 2.57% in terms of ROUGE-1, ROUGE-2, and ROUGE-L, respectively. Compared to `TitleGen-FL` with only the IR module, `TitleGen-FL` can improve its performance by 8.80%, 13.69%, and 8.86% in terms of ROUGE-1, ROUGE-2, and ROUGE-L, respectively. Finally, based on our human study, we also verify the effectiveness of our proposed filter `TitleGen-FL` and it is reasonable to use both the DL module and the IR module in our proposed quality prediction-based filter.

To our best knowledge, the main contributions of our study can be summarized as follows.

- To make the state-of-the-art issue title generation method iTAPE (Chen et al., 2020) more practical, we propose a quality prediction-based filter `TitleGen-FL`. Specifically, `TitleGen-FL` contains the DL module and the IR module. The DL module can automatically learn the features from a large-scale training set after analyzing the quality of the titles generated by iTAPE. While the IR module is based on whether there are historical issues that are sufficiently similar to the new issue.
- We conduct experiments on the benchmark dataset gathered from real-world open-source projects on GitHub. Both the automatic evaluation and the human study can verify the effectiveness of our proposed quality prediction-based filter and the rationality of using both modules in `TitleGen-FL`.
- Based on our proposed filter, we developed a plug-in to assist developers to generate the issue title by analyzing the contents of the issue body. Moreover, to facilitate other researchers to replicate our study, we also share our replication package.[3]

The rest of this paper is structured as follows. Section 2 illustrates the motivation of our study. Section 3 shows the framework of our proposed filter, the details of each component, and our developed tool. Section 4 introduces our designed research questions, performance measures, implementation details. Section 5 performs result analysis on our designed research questions. Section 6 first analyzes the main reasons, which make iTAPE (Chen et al., 2020) cannot generate high ROUGE-L values. Then this section analyzes the performance influence of different DL models (such as TextCNN, TextRNN, and Transformer) on the DL module. Section 7 discusses potential threats to the validity of our empirical study. Section 8 analyzes the related studies and emphasizes the novelty of our study. Section 9 concludes this study and discusses potential future work.

2. Research motivation

To help developers generate high-quality titles for the issues on GitHub, Chen et al. (2020) were the first to formulate this problem into a one-sentence summarization problem. Then they proposed the method iTAPE based on a seq2seq model (Sutskever et al., 2014). Moreover, they adopted the copy mechanism (Gu et al., 2016; See et al., 2017) and proposed the human-named token tagging method, which can alleviate the low term frequency problem of identifiers and version numbers in the issue body.

Their empirical study showed that the method iTAPE can achieve promising performance. However, we made an in-depth analysis of the distribution of ROUGE-L score distribution after applying the method iTAPE (Chen et al., 2020) to the testing set, and the final analysis result can be found in Fig. 1. In this figure, the *x*-axis denotes ROUGE-L score ranges and the *y*-axis denotes the corresponding proportion of titles. According to the statistical results, we can find the ROUGE-L scores of about 57.3% of titles are less than 0.3.

Then we analyze the main reasons for the low ROUGE-L score in a manual way. Here we show the top-3 reasons and their

3 https://github.com/TitleGen-FL/TitleGen-FL

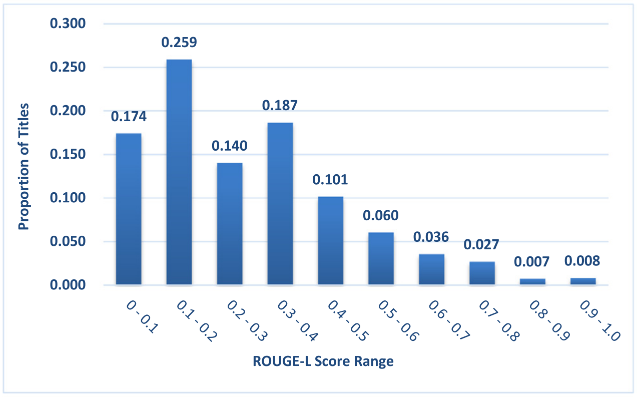


Fig. 1. ROUGE-L score distribution of the generated titles when applying the method iTAPE to the testing set for our experimental subject.

Ground-truth Title: Repeat saving unmodified files triggers Eg webpack compilation then running slowly
Generated Title: vscode doesn't change file info
Issue Body:
(https://user-images.Github.usercontent.com/3961388/65826287-bf1f4900-e2b4-11e9-9583-35eca9972632.gif)
A picture is worth a thousand wordsI often press `⌘ + s` because I want to auto format, but VSCode always change file info even I have not modified the file content, then triggers Eg webpack compilation then running slowly.😥 And Webstorm will not modify the modification time information of the file, could I configure it?
URL: https://github.com/microsoft/vscode/issues/81595

Fig. 2. In case A, the generated title is low-quality in this case.

corresponding examples in Fig. 2, Fig. 3, and Fig. 4. Each example includes the ground-truth title written by the developer, the title generated by iTAPE, the issue body, and the URL in GitHub. A more detailed analysis of these reasons can be found in Section 6.1.

In Case A, the quality of the generated title is not high, and it cannot provide a concise and precise summarization of the issue body like the ground-truth title. In this case, we classify the reason as **the low quality of the generated title**.

In Case B, we find after using the preprocessing steps provided by iTAPE, some important information is lost. Since some issue bodies contain URLs, images, code snippets, and lines that start with unchecked Markdown checkboxes, the method iTAPE used regular expressions to replace these types of non-textual information with placeholders. In this case, the preprocessed issue body becomes "a.rs : phofnewline phofnewline phofcode phofnewline phofnewline b.rs : phofnewline phofnewline phofcode phofnewline phofnewline c.rs : phofnewline phofnewline phofcode phofnewline phofnewline ran ' rustc a.rs ', then ' rustc b.rs -l. ' . the result is a confusing error : phofnewline phofnewline phofcode phofnewline phofnewline rustc verid40 0.11-pre-nightly verid0 (d35804e 2014-04-18 00:01:22 -0700) phofnewline" and we cannot find enough useful information from this text. Obviously, the method iTAPE cannot generate a high-quality title based on this preprocessed text. Notice that this problem also exists in the original issue body (i.e., the developer does not provide enough useful information when writing the issue body). In this case, we classify the reason as **the low-quality issue body**.

Ground-truth Title: confusing error with 'extern crate'
Generated Title: confusing error when compiling rustc
Issue Body:
a.rs:
```
#![crate_type = "dylib"]
pub trait Foo {}
```
b.rs:
```
mod c;
fn main() {}```
c.rs:
```
extern crate a;
use a::Foo;
~~~
Ran `rustc a.rs`, then `rustc b.rs -L.`. The result is a confusing error:
```c.rs:2:9: 2:10 error: unresolved import. maybe a missing `extern crate`? c.rs:2 use a::Foo;```
rustc 0.11-pre-nightly (d35804e 2014-04-18 00:01:22 -0700)
**URL:** https://github.com/rust-lang/rust/issues/13672

**Fig. 3.** In case B, the issue body after data preprocessing is low-quality in this case.

**Ground-truth Title:** 500 when should be 404 error message
**Generated Title:** 500 error on 404 page
**Issue Body:**
Don't know if you already know but commit`bb17e1c0e9a4bf93df059d4c40a9088f5e71c47d` broke the default error handeling. For every 404 page it is showing a 500 message with `[object][object]` message.
I have tested it with commit`dee054e2c3931135f1c63b3becd01333af99800b` and that is still showing proper 404 messages. Commit`bb17e1c0e9a4bf93df059d4c40a9088f5e71c47d` shows 500 messages.
Looks like the changes in `core/server/api/index.js` are the problem here.
**URL:** https://github.com/TryGhost/Ghost/issues/1466

**Fig. 4.** In case C, the generated title with a similar semantic but dissimilar description in this case.

In case C, the ground-truth title is "500 when should be 404 error message" and the generated title is "500 error on 404 page". In this case, we can find that the generated title has a similar semantic to the ground-truth title. However, the descriptions of these two titles are quite different, which results in the low ROUGE-L value. In this case, we classify the reason as **the generated title with a similar semantic but dissimilar description**. Notice this reason is not due to the poor quality of the generated title, but due to the limitations of the currently used performance measures (such as BLEU (Papineni et al., 2002), ROUGE (Lin, 2004)), which only measure the quality at the lexical level while not at the semantic level.

Solving these problems is a challenging task for the issue title generation problem. For the problem of the low quality of the generated title, we should improve the method iTAPE with more high-quality training data or with more advanced modeling methods. For the problem of the low-quality issue body, if the
~~~

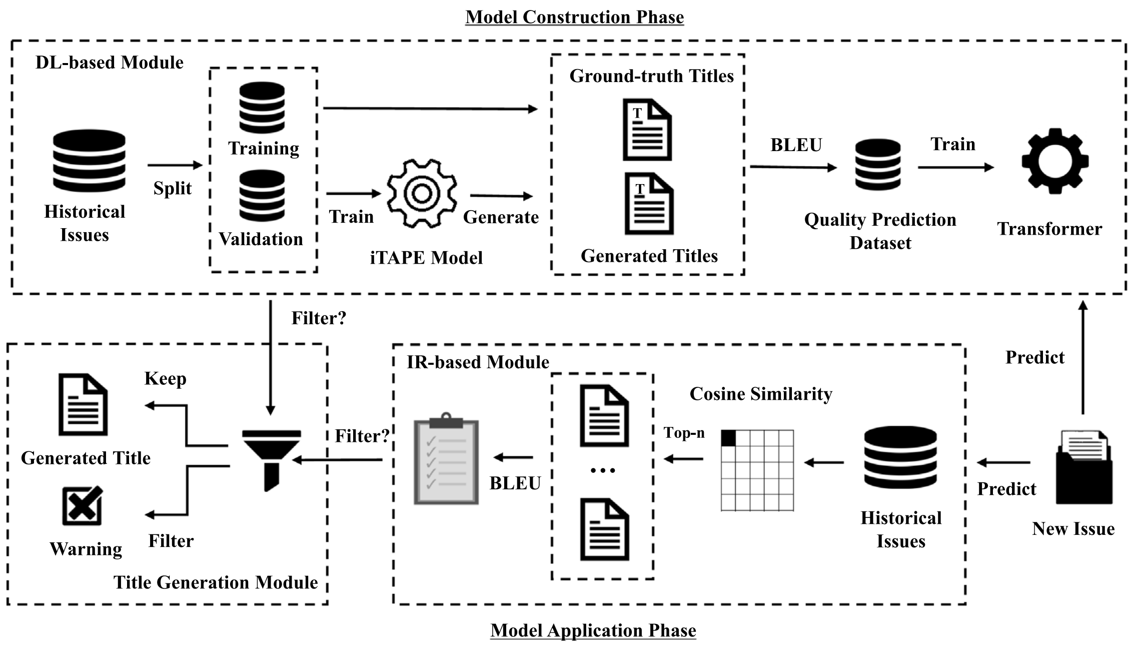


Fig. 5. Framework of our proposed filter `TitleGen-FL`.

quality problem exists in the original issue body, we should provide feedback to developers and ask them to provide more useful information. If the quality problem exists in the preprocessed issue body, we should design effective preprocessing methods, which can avoid important information loss during the preprocessing process. For the problem of the generated title with a similar semantic but dissimilar description, we should design more effective performance measures, since currently used measures (such as BLEU (Papineni et al., 2002), ROUGE (Lin, 2004)) cannot solve this problem. Different from the above possible solutions, we aim to solve these problems from another perspective (Jiang et al., 2017a; Hu et al., 2021; Wang et al., 2021). In our study, we want to design a quality prediction-based filter to automatically filter out issues, which are difficult to generate high-quality titles. Therefore, our solution can help to make the state-of-the-art issue title generation method iTAPE (Chen et al., 2020) more practical.

3. Our proposed filter

In this section, we first show the framework of our proposed filter `TitleGen-FL`. Then we introduce the details of each module in `TitleGen-FL`.

3.1. Framework of `TitleGen-FL`

The framework of `TitleGen-FL` can be found in Fig. 5. In particular, the core of `TitleGen-FL` includes two modules: a deep learning-based (DL) module and an information retrieval-based (IR) module. In the title generation module, if either module predicts that the high-quality titles cannot be generated, `TitleGen-FL` can automatically filter these issues and give warning messages. Otherwise, the issue title generated by the method iTAPE is returned. Notice there exists some data preprocessing difference for these two modules. Specifically, for the IR module, we simply represent the issue body based on TF–IDF. For the deep learning module, we utilize data preprocessing operations (such as word segmentation and length fix) provided by torchtext framework.[4]

[4] https://pypi.org/project/torchtext

3.2. Deep learning-based module

In Section 2, we mainly analyze the main reasons for the low ROUGE-L score in a manual way. However, it is difficult to enumerate all patterns or implement them into rules. Recently, the rise of deep learning (Goodfellow et al., 2016) provides a way to automatically learn the features from large-scale training data. Therefore, we formulate the issue filter as the classification task and adopt the deep learning model to construct the prediction model.

Given the training set D_{Train} and the validation set D_{Val}, the DL module constructs the classification model as follows.

- **Step 1:** The DL module trains the generation model M_{Gen} via the method iTAPE (Chen et al., 2020) based on D_{Train} and D_{Val}.
- **Step 2:** The DL module uses the trained model M_{Gen} to generate titles for the issues in $D_{Train} \cup D_{Val}$.
- **Step 3:** The DL module computes the ROUGE-L score based on the generated title and the ground-truth title for the issues in $D_{Train} \cup D_{Val}$.
- **Step 4:** The DL module constructs the quality prediction dataset D_{Qua}. For each issue in $D_{Train} \cup D_{Val}$, we first select its issue body as the input, then assign its label as 1 if its ROUGE-L score is larger than the threshold value and assign its label as 0 if its ROUGE-L score is smaller than the threshold value. In this study, we set the threshold value to 0.3 according to our manual analysis results. In particular, since we cannot manually analyze the title quality for all the issues in the testing set (i.e., 33,438 issues), we randomly selected 100 issues for different ROUGE-L score ranges from the testing set and manually analyzed the quality of these issue titles. We find when the ROUGE-L score is lower than 0.3, the title quality decreases rapidly. Moreover, we use the simple random under-sampling strategy to solve the class imbalanced problem in D_{Qua} (i.e., removing the instances in the majority class until reaching the number of instances in the minority class).
- **Step 5:** The quality prediction dataset D_{Qua} is split into 70%: 30% as the training set and the validation set. Then the DL module trains the classification model M_{Class} via the

DL model Transformer (Vaswani et al., 2017). The Transformer is the first transduction model relying on the attention mechanism and has shown competitive performance on different tasks (such as machine translation (Wang et al., 2019), source code summarization (Ahmad et al., 2020; Li et al., 2022; Yang et al., 2022a, 2021a), code completion (Liu et al., 2020a), query reformulation (Cao et al., 2021), Stack Overflow title generation (Liu et al., 2022)).

For a new issue, the DL module extracts its issue body and then uses the trained model M_{Class} to predict whether the method iTAPE can generate the high-quality title.

3.3. Information retrieval-based module

For the issue title generation task, if there are no historical issues that are sufficiently similar to the new issue, it may be difficult for the method iTAPE to generate a high-quality title. Therefore, the IR module computes the IR score based on the similarity between its corresponding issue body and the issue bodies in historical issues. The design of the IR module is mainly inspired and followed by the commit message generation method NNGen (Liu et al., 2018), which is a lightweight nearest neighbor-based method and can outperform the high-weight deep learning-based method NMT (Jiang et al., 2017a). In particular, for a new issue, the process of the IR module can be described as follows.

- **Step 1:** The IR module represents the issue bodies in $D_{train} \cup D_{Val}$ as the TF–IDF (Term Frequency–Inverse Document Frequency) vectors.
- **Step 2:** The IR module represents the issue body of the new issue as a TF–IDF vector.
- **Step 3:** The IR module computes the cosine similarity between the new issue and the historical issues in $D_{train} \cup D_{Val}$ based on their TF–IDF representation. Then this module selects top-n historical issues with the largest similarities. In our study, we set the value of n to 5.
- **Step 4:** The IR module returns the IR score as

  $$max_{j\in\{1,\ldots,n\}}\{BLEU(issue_{new}, issue_{top\text{-}j})\} \quad (1)$$

  where $issue_{top\text{-}j}$ denotes the historical issue with the jth highest similarity to the new issue $issue_{new}$ and the similarity is measured by BLUE performance measure (Papineni et al., 2002) by following NNGen (Liu et al., 2018) due to its popularity in source code summarization. In this module, we set the threshold value to 0.1 based on our manual analysis results. This means when the IR score is less than 0.1, the IR module will predict that the method iTAPE cannot generate the high-quality title. Notice, we can also use ROUGE-L to measure the issue similarity since these two performance measures are all designed based on the n-gram overlap.

Compared to the cosine similarity, the BLEU score can further take into account the word order of the issue body. However, the computational cost of the BLEU score is high. To speed up the IR module, we do not find the nearest neighbors in terms of BLEU. Instead, we first use the cosine similarity to find the top-n historical issues with the largest similarities. Then we select the best issue from these n historical issues in terms of BLEU. Therefore, the setting of the IR module can balance the computational cost and the performance of the IR module.

Table 1
Statistics of the experimental subject.

Dataset	Issues	Issue body			Issue title		
		Mean	Max.	Min.	Mean	Max.	Min.
Training	267,094	126.61	1,150	30	9.10	38	5
Validation	33,031	126.01	746	30	9.10	36	5
Testing	33,438	126.71	720	30	9.11	33	5

3.4. Our developed tool

To help developers generate high-quality titles, we developed a plug-in based on iTAPE (Chen et al., 2020) and our proposed filter `TitleGen-FL`. Therefore, during the software development and maintenance, when the developers craft the issue, they can focus on the writing of the issue body. The screenshot of our developed plug-in can be found in Fig. 6. If our developed plug-in predicts that it is difficult to generate a high-quality title, it can return the alert information (in Fig. 6(a)). Otherwise, the plug-in can provide the candidate title generated by iTAPE (in Fig. 6(b)).

4. Experimental setup

In this section, we want to answer the following three research questions:

RQ1: How effective is our proposed filter `TitleGen-FL` when applied to the state-of-the-art issue title generation method iTAPE?

RQ2: How effective are the DL module and the IR module in our proposed filter `TitleGen-FL` for automated issue title generation?

RQ3: Based on the human study, can our proposed filter `TitleGen-FL` effectively preserve the issues, which can generate high-quality titles? Do we need to use the DL module and the IR module at the same time in our proposed filter `TitleGen-FL`?

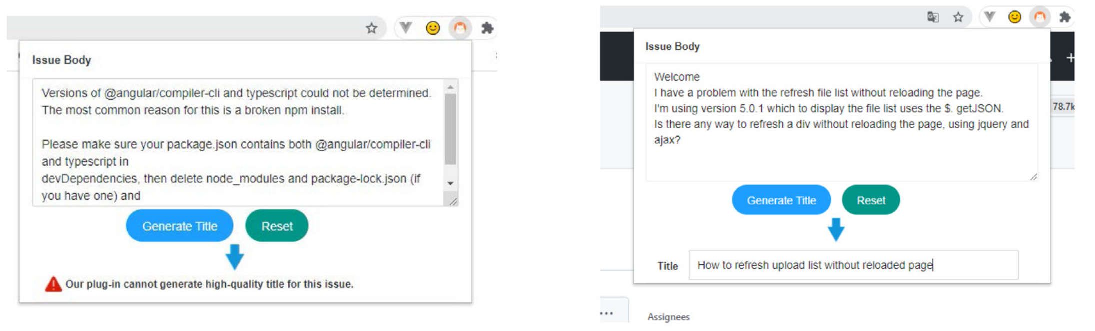
To answer these three RQs, we first introduce our used experimental subject. Then we illustrate our considered performance measures. Later, we show our automatic evaluation and human evaluation methods. Finally, we give the implementation details.

4.1. Experimental subject

In this study, we select the dataset shared by Chen et al. (2020) as our experimental subject. They selected issues from the repositories in GitHub with the top 200 number of stars and originally collected 922,730 issues. Then they designed three heuristic rules to filter issues, which have low-quality titles or do not fit the context of the issue title generation task. Their considered heuristic rules are:

- **Rule 1:** Filter the issues, whose titles contain less than 5 words, or more than 15 words, or contain URLs.
- **Rule 2:** Filter the issues, whose titles have more than 70% words missing in the issue bodies.
- **Rule 3:** Filter the issues, whose title has a sub-sequence that can exactly match a specific part of the issue body while the sub-sequence is over 70% of the title length.

After using these heuristic rules, they finally obtained 333,563 issues. These issues are split into 80%:10%:10% as the training set, the validation set, and the testing set. The statistics of this experimental subject can be found in Table 1. These statistics include the number of the issues, the min, max and mean length of the issue bodies, and the issue titles respectively. In this table, we can find the average number of words in the issue body and issue title is 126 and 9 respectively.


Issue Body

Versions of @angular/compiler-cli and typescript could not be determined. The most common reason for this is a broken npm install.

Please make sure your package.json contains both @angular/compiler-cli and typescript in devDependencies, then delete node_modules and package-lock.json (if you have one) and

Our plug-in cannot generate high-quality title for this issue.

Issue Body

Welcome
I have a problem with the refresh file list without reloading the page.
I'm using version 5.0.1 which to display the file list uses the $. getJSON.
Is there any way to refresh a div without reloading the page, using jquery and ajax?

Title How to refresh upload list without reloaded page

(a) In this scenario, our developed plug-in predicts that it is difficult for the method iTAPE to generate the high-quality title

(b) In this scenario, our developed plug-in predicts that the method iTAPE can generate the high-quality title and return its generated title.

Fig. 6. Screenshot of our developed plug-in.

4.2. Performance measures

To evaluate the quality of the generated titles, we consider the performance measure ROUGE, which is also used in a previous study for issue title generation (Chen et al., 2020). ROUGE (Recall-Oriented Understudy for Gisting Evaluation) (Lin, 2004) is similar to BLEU. Specifically, ROUGE-N measures the overlap of *n*-grams between the generated titles and the reference titles. While ROUGE-L takes into account sentence-level structure similarity and automatically identifies the longest co-occurring in the sequence *n*-grams. To ensure the implementation correctness of these performance measures, we use ROUGE package[5] to compute the values of ROUGE.

Table 2
Hyper-parameters and their values of the transformer model.

Category	Hyper-parameter	Value
Model structure	n_layers	2
	n_heads	5
	d_model	300
	hidden_size	1024
	max input length	100
Model training phase	dropout	0.5
	optimizer	Adam
	learning rate	0.001
	batch size	256
	activation function	Relu

4.3. Evaluation methods

4.3.1. Automatic evaluation

The intuitive method to evaluate the performance of our proposed `TitleGen-FL` is to compute the performance measures (such as precision and recall) based on the confusion matrix. However, the performance computation depends on whether each generated title is semantically relevant to the ground-truth title, which needs a huge manual label cost. In this study, we use the overall quality, which is the average value of all the considered titles measured by the specific performance measure (i.e., ROUGE-1, ROUGE-2, or ROUGE-L). If the value is higher than that of the generated titles without using the filter `TitleGen-FL`, we can conclude our proposed filter `TitleGen-FL` can effectively filter the issues, which cannot generate high-quality titles. We notice the comparison between `TitleGen-FL` and iTAPE may be misleading since distinct issue sets are used to compute the evaluation measures. Therefore we consider a new method, which randomly filters the issues with the same filter ratio and then measures the overall quality of the preserved titles.

4.3.2. Human evaluation

The above performance measures may not correlate reasonably well with the quality of the generated titles, which is discussed in Section 2. Therefore, we hire six master students to conduct a human study. These students are not on the list of paper authors. They are all from the software engineering major and have rich experience in using GitHub for software development and maintenance. We first randomly select 200 issues from the testing set and randomly divide them into two equal groups. Then we also randomly divide these six students into two equal groups. Therefore, three students in each group are asked to evaluate 100 issues. Later, these students are asked to give a quality score between 1 and 5 after reading the issue body, the title generated by iTAPE, and the ground-truth title. The quality score can evaluate the semantic similarity between the generated title and the ground-truth title. Here the score of 1 means there is no semantic relevance between these two titles, while the score of 5 means these two titles have the same semantics. For each generated title of the issue, we can obtain three quality scores from three different students. Therefore, We obtain the final quality score as the average of these three scores. In our study, we consider the final quality score between 4 and 5 for the generated title to the high quality, and the remaining final quality scores to be low quality. Notice that for our human study, the evaluation is done based on documents. During the evaluation, the participants cannot see the scores given by others. For a fair comparison, we also shuffled the order of the issues. Finally, to avoid the errors caused by long-term annotation, we limited the number of issues that the hired students can evaluate in a fixed time.

4.4. Implementation details

For the method iTAPE, since they did not share the trained issue title generation model, we directly used the script shared by Chen et al. (2020) to retrain the model, and the training process was performed based on the same experimental setting. Although it exists a performance difference from the results reported in their published paper, the performance difference is not significant. For our proposed filter `TitleGen-FL`, we mainly use the Transformer algorithm provided by Pytorch 1.6.0 to implement the DL module. The hyper-parameters of the Transformer model are optimized and the final values can be found in Table 2.

We run our experiments on a computer with an Intel(R) Core(TM) i7-8700 CPU and a GeForce RTX3090 GPU (40 GB memory). The operating system is Windows 10.

5 https://pypi.org/project/rouge

Table 3
Effectiveness of TitleGen-FL when applied to iTAPE based on the automatic evaluation.

Method	ROUGE-1	ROUGE-2	ROUGE-L	Filter ratio
iTAPE	0.292	0.126	0.281	0.00%
iTAPE+Random	0.291	0.126	0.279	56.59%
iTAPE+TitleGen-FL	**0.324**	**0.151**	**0.312**	56.59%

5. Result analysis

5.1. Result analysis for RQ1

In RQ1, we want to show the competitiveness of our proposed TitleGen-FL in terms of the ROUGE performance measure. To our best knowledge, the method iTAPE was the first to study the issue title generation problem on GitHub. Therefore, we mainly consider iTAPE as our baseline. Specifically, we first use the method iTAPE to measure the overall quality (i.e., average value) of the titles generated for the issues in the testing set. We second apply TitleGen-FL (i.e., iTAPE+TitleGen-FL) to filter out the issues, which cannot generate high-quality titles. Then we show the ratio of the filtered issues to all the issues in the testing set (i.e., filter ratio) and measure the overall quality of the titles preserved by TitleGen-FL. Finally, to guarantee the comparison fairness, we also consider a baseline (i.e., iTAPE+Random), which is introduced in Section 4. To avoid the influence of the random factors in this baseline, we repeat this method 10 times and take the average performance value.

The comparison results of these three methods in terms of ROUGE-1, ROUGE-2, and ROUGE-L can be found in Table 3. In this table, we can find our proposed filter can automatically filter 56.59% of issues. For the preserved issues, TitleGen-FL can achieve 0.324, 0.151, and 0.312 in terms of ROUGE-1, ROUGE-2, and ROUGE-L. Compared to the method iTAPE, TitleGen-FL can improve the performance by 10.87%, 19.47%, and 11.22% in terms of ROUGE-1, ROUGE-2, and ROUGE-L. Compared to the method iTAPE+Random, TitleGen-FL can improve the performance by 11.34%, 19.79%, and 11.92% in terms of ROUGE-1, ROUGE-2, and ROUGE-L.

Summary for RQ1: Based on the automatic evaluation, TitleGen-FL can effectively filter the issues, which cannot generate high-quality titles when using the method iTAPE.

5.2. Result analysis for RQ2

In RQ2, we want to conduct an ablation study to demonstrate the effectiveness of the two components used in our proposed TitleGen-FL. We first use the Venn diagram to analyze the diversity of the preserved issues for different modules. The final results can be found in Fig. 7. Specifically, when only using the DL module, this filter can preserve 22,553 issues. When only using the IR module, this filter can preserve 21,611 issues. Moreover, we can find these two modules have a certain diversity since only 14,516 issues can be preserved when using both two modules.

Then we further compare TitleGen-FL with two controlled methods. The first controlled method (i.e., iTAPE+DL) only uses the DL module to filter the issues and the second controlled method (i.e., iTAPE+IR) only uses the IR module to filter the issues. The comparison results of our ablation study can be found in Table 4. Compared to iTAPE+DL, TitleGen-FL can improve its performance by 2.31%, 5.61%, and 2.57% in terms of ROUGE-1, ROUGE-2, and ROUGE-L. Compared to iTAPE+IR, TitleGen-FL can improve its performance by 8.80%, 13.69%, and 8.86% in terms of ROUGE-1, ROUGE-2, and ROUGE-L.

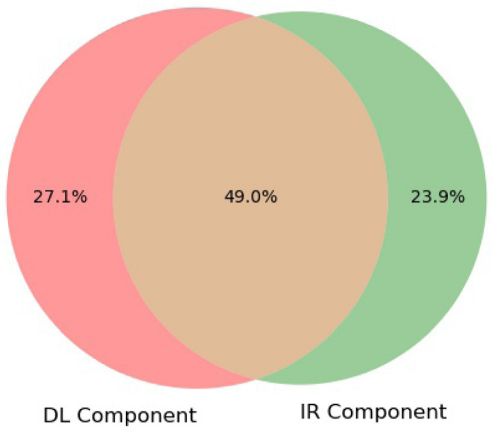


Fig. 7. The diversity on the preserved issues for different modules.

Table 4
The comparison results of the ablation study based on the automatic evaluation.

Method	ROUGE-1	ROUGE-2	ROUGE-L	Filter ratio
iTAPE+DL	0.317	0.143	0.304	32.55%
iTAPE+IR	0.298	0.133	0.287	35.37%
iTAPE+TitleGen-FL	**0.324**	**0.151**	**0.312**	56.59%

Summary for RQ2: Based on the automatic evaluation, by using both the DL module and the IR module, TitleGen-FL can help to effectively preserve the issues, which can help to generate high-quality titles when using the method iTAPE.

5.3. Result analysis for RQ3

In RQ3, we want to show the competitiveness of TitleGen-FL through human study. We first analyze whether our proposed filter TitleGen-FL can effectively preserve the issues, which can generate high-quality titles based on the human study. The human study result on 200 issues can be found in Table 5. Here the columns **1** to **5** show the distribution of the final quality score by hired master students, and the column **# Preserved** shows the number of the preserved issues (Notice that iTAPE does not filter any issues and the value of **# Preserved** is 200), and the column **Mean Score** represents the mean scores of the final quality scores on preserved issues. In this table, we can find using TitleGen-FL can preserve 84 issues (i.e., 42.00% of the sampled issues). Compared to the method iTAPE and the method iTAPE+Random, our proposed filter can preserve the issues with more high-quality generated titles both in quality score distribution and the mean score.

We also compare TitleGen-FL with the random method TitleGen-FL+Random in terms of the performance measures based on the confusion matrix. Here, we use HT to denote the issue, whose title is generated by iTAPE and the evaluated quality score from three students is between 4 and 5. Then we use LT to denote the issue, whose title is generated by iTAPE and the evaluated quality score from three students is less than 4. If we treat the HT issues as the positive instances and the LT issues as the negative instances, we can classify the sampled issues into four types according to the actual type (evaluated and determined by students) and the predicted type (evaluated and determined

Table 5
Effectiveness of TitleGen-FL when applied to iTAPE based on the human study.

Method	1	2	3	4	5	# Preserved	Mean score
iTAPE	8.00%	8.00%	16.00%	37.00%	31.00%	200	3.750
iTAPE+Random	5.95%	8.33%	13.10%	39.29%	33.33%	84	3.857
iTAPE+TitleGen-FL	1.19%	2.38%	11.90%	46.43%	38.10%	84	**4.179**

Table 6
Effectiveness of TitleGen-FL in terms of precision and recall when applied to iTAPE based on the human study.

Method	$Precision_{HT}$	$Recall_{HT}$	# Preserved	# HT	$Precision_{LT}$	$Recall_{LT}$	# Filtered	# LT
iTAPE+Random	0.726	0.449	84	136	0.353	0.641	116	64
iTAPE+TitleGen-FL	0.845	0.522			0.440	0.797		

Table 7
The comparison results of the ablation study based on the human study.

Method	1	2	3	4	5	# Preserved	Mean score
iTAPE+DL	5.22%	5.97%	14.93%	39.55%	34.33%	134	3.918
iTAPE+IR	3.97%	5.56%	12.70%	42.86%	34.92%	126	3.992
iTAPE+TitleGen-FL	1.19%	2.38%	11.90%	46.43%	38.10%	84	**4.179**

by TitleGen-FL): true positive, false positive, true negative, and false negative. We use TP, FP, TN, and FN to denote the number of true positives, false positives, true negatives, and false negatives, respectively. Then the performance measure $Precision_{HT}$ can be computed as follows.

$$Precision_{HT} = \frac{TP}{TP + FP} \tag{2}$$

The performance measure $Recall_{HT}$ can be computed as follows.

$$Recall_{HT} = \frac{TP}{TP + FN} \tag{3}$$

Here, a higher value of $Precision_{HT}$ means more issues that can generate high-quality titles by iTAPE can be preserved.

If we treat the LT issues as the positive instances and the HT issues as the negative instances, we can use the same way to compute the two performance measures $Precision_{LT}$ and $Recall_{LT}$. Here, a higher value of $Recall_{LT}$ means more issues that cannot generate high-quality titles by iTAPE can be filtered.

The final comparison result can be found in Table 6. Here, the column **# HT** denotes the number of the HT issues in our sampled issues while the column **# LT** denotes the number of the LT issues in our sampled issues. The column **# Preserved** denotes the number of the preserved issues while the column **# Filtered** denotes the number of the filtered issues. In this table, we can find that our proposed filter TitleGen-FL can still outperform the random method in terms of these performance measures, since TitleGen-FL can achieve higher $Precision_{HT}$ and higher $Recall_{LT}$.

In RQ3, we second analyze whether we need to use the DL module and the IR module at the same time in our proposed filter TitleGen-FL. The human study result can be found in Table 7. In this table, we can find our proposed filter can preserve the issues with more high-quality generated titles both in quality score distribution and the mean score after comparing with the method only using the DL module or only using the IR module.

We also analyze the result of the ablation study in terms of the performance measures based on the confusion matrix. The final comparison result can be found in Table 8. Notice in our proposed filter, if either of these two modules predict that the high-quality title cannot be generated by iTAPE, TitleGen-FL can automatically filter this issue and return a warning message. Therefore, compared to iTAPE+DL and iTAPE+IR, our proposed filter certainly preserve fewer issues and filter more issues. Higher $Precision_{HT}$ and higher $Recall_{LT}$ means the preserved issues has a higher probability of containing issues, which can generate high-quality titles. However, there also exist some issues, which can generate high-quality titles, were filtered by our proposed filter. First, this is because we only use the ROUGE-L value to measure the quality of the title, which can mislabel some issues when the generated title and the ground-truth have high semantic similarity. Second, the ground-truth title quality provided by developers may also not high. These problems are still highly challenging and need further studies to alleviate them.

Summary for RQ3: Based on our human study, we can further verify the effectiveness of our proposed filter TitleGen-FL. Moreover, we can still find it is reasonable to use both the DL module and the IR module at the same time in our proposed filter.

6. Discussions

In this section, we first analyze why these issues are filtered by TitleGen-FL. Then we analyze the performance influence of the DL Module when considering different DL models.

6.1. Why these issues are filtered by TitleGen-FL

In this section, we analyze the main reasons, which make iTAPE (Chen et al., 2020) cannot generate high-quality titles. Due to the high cost of manually analyzing all the filtered issues in the testing set, we use a commonly-used sampling method (Singh and Mangat, 2013) to select the minimum issues for analysis. The number of the sampled issues can be computed as follows.

$$MIN = \frac{n_0}{1 + \frac{n_0 - 1}{size}} \tag{4}$$

where n_0 depends on the confidence level and the desired error margin $n_0 \left(= \frac{Z^2 \times 0.25}{e^2}\right)$. Z is a confidence level z score and e is the error margin. *size* is the number of issues in the testing set. In this study, we select *MIN* issues with the error margin $e = 0.05$ at 95% confidence level (i.e., $MIN = 366$). During the analyzing process, we first identify the five main reasons. Then we ask two master students to label the reason for each issue. If there are differences in the labeled results, they will be asked to discuss them to reach a consensus.

The reason and its corresponding proportion can be found in Table 9. In this table, we can find the top-3 reasons that account

Table 8
The result of the ablation study in terms of precision and recall based on the human study.

Method	$Precision_{HT}$	$Recall_{HT}$	# Preserved	# HT	$Precision_{LT}$	$Recall_{LT}$	# Filtered	# LT
iTAPE+DL	0.739	0.728	134	136	0.439	0.453	66	64
iTAPE+IR	0.778	0.721	126		0.486	0.563	74	
iTAPE+`TitleGen-FL`	0.845	0.522	84		0.440	0.797	116	

Table 9
Distribution of different reasons.

ID	Reason	Proportion
1	The low quality of the generated title	52.4%
2	The low-quality issue body	28.2%
3	The generated title with similar semantic but dissimilar description	10.6%
4	The low quality of the ground-truth title	8.2%
5	Wrong filtered	0.6%

Table 10
The performance of `TitleGen-FL` when the DL module considers different DL models.

DL Model	ROUGE-1	ROUGE-2	ROUGE-L	Filter ratio
TextCNN	0.309	0.140	0.297	43.55%
TextRNN	0.306	0.139	0.295	44.76%
RCNN	0.307	0.139	0.296	44.01%
RNN-Attention	0.319	0.147	0.307	52.34%
Transformer	**0.324**	**0.151**	**0.312**	56.59%

for the top 91.2%. These reasons and corresponding cases can be found in Section 2. Except for these three reasons, there are other two reasons. Specifically, the first reason is the low quality of the ground-truth title, which accounts for 8.2%. This means the quality of the used dataset needs to be improved in the future. The second reason is that the issues are wrongly filtered, which accounts for 0.6%. This means our proposed filter still has the performance improvement room.

6.2. Influence of different DL models on DL-based module

In this subsection, we mainly analyze the influence of the DL module on the performance of `TitleGen-FL` when considering different DL models. Here, we consider the other four classical DL models and briefly introduce these models as follows.

- **TextCNN** (Kim, 2014). TextCNN is based on a convolutional neural network and is used to extract information similar to *n*-gram in sentences.
- **TextRNN** (Liu et al., 2016). TextRNN is based on bi-directional LSTM and can capture di-directional *n*-gram information with variable length.
- **TextRNN+Attention** (Luong et al., 2015). Based on the TextRNN model, this method further adds the attention mechanism, which can alleviate the long-term dependence problem of the text.
- **RCNN** (Lai et al., 2015). RCNN (Recurrent Convolutional Neural Network) applies a recurrent structure to capture contextual information when learning word representations. Moreover, RCNN utilizes a max-pooling layer, which can automatically determine which words play key roles in the text classification.

In our study, we also use Pytorch 1.6.0 to implement these deep learning methods and follow the setting used in Transformer. The comparison result can be found in Table 10. In this table, we can find that among these DL models, when the DL module considers the DL model Transformer, `TitleGen-FL` can achieve the best performance. Specifically, based on the similar filter ratio, using Transformer can at least improve the performance by 1.65%, 2.49%, and 1.63% in terms of ROUGE-1, ROUGE-2, and ROUGE-L, respectively.

7. Threats to validity

In this section, we mainly discuss potential threats to the validity of our empirical study.

Internal threats. The main internal threat concerns potential faults in the implementation of our proposed filter `TitleGen-FL` and the issue title generation method iTAPE. To alleviate this threat, we use mature frameworks and software testing to guarantee the code implementation correctness. For the method iTAPE, we use the script provided by Chen et al. (2020) to train the generation model with the same experimental setting.

External threats. The main external threat is the limited number of investigated repositories on GitHub. To alleviate this threat, we select the dataset shared by Chen et al. (2020) as our experimental subject. In this dataset, they considered the top 200 starred repositories in GitHub, since these repositories are generally well-maintained.

Construct threats. The first construct threat is the bias of our human study. To alleviate this threat, We first hire master students, which are familiar with the usage of GitHub. Then we provided a tutorial before our human study to ensure all the master students comprehend our protocol. Finally, we use Fleiss Kappa (Fleiss, 1971) to measure the agreement among the students in two groups. The Kappa scores are 0.625 and 0.619 respectively, which indicates substantial agreement in these two groups. Then the second construct threat is the performance measures used in our automatic evaluation. In our study, we consider the performance measures ROUGE, which can measure the text similarity between the generated title and the ground-truth title. Moreover, this performance measure has been widely used in previous studies on the issue title generation (Chen et al., 2020) and similar problems (such as source code summarization (Wan et al., 2018; Hu et al., 2018; LeClair et al., 2019; Yang et al., 2021b, 2022b; Li et al., 2021)). The third construct threat is the threshold value for ROUGE-L used to determine whether the quality of the generated title is high or low. To alleviate this threat, we set the threshold value for ROUGE-L to 0.3 by our manual analysis and can achieve promising results in our empirical study. However, this performance measure is computed on n-gram overlap (Haque et al., 2022), we will investigate other performance measures, which can reflect the semantic difference between the generated title and the ground-truth title.

8. Related work

In modern software development and maintenance, an accurate and concise summary of the bug report can help developers understand the bug reports. Bug report summarization is a promising method of automated generating summaries by selecting salient sentences.

In the early studies, researchers mainly focused on supervised methods. For example, Rastkar et al. (2014) investigated whether existing conversation-based automated summarizers were applicable to bug report summarization. Then Jiang et al. (2017b) leveraged the authorship characteristics of contributors for bug report summarization. However, these supervised methods need huge label efforts for preparing the training set (Lotufo et al., 2015). Therefore, some researchers started to focus on unsupervised methods. For example, Li et al. (2018) proposed the method DeepSum. This method is based on a stepped auto-encoder network, which infers the bug report summary based on the hidden layers of the network. Then this method further incorporates the evaluation enhancement module and the predefined field enhancement module. Recently, Liu et al. (2020b, 2021) proposed an unsupervised method BugSum. This method first used a trained auto-encoder network to extract semantic features. Then this method designed a novel metric to measure the degree that a sentence has been approved against disapproval in the interactive discussions. Finally, this method used a dynamic selection strategy based on informativeness and comprehensiveness to optimize the summarization process.

It is not hard to find that the previous studies focused on the bug reports in bug tracking systems (such as Bugzilla). While we focus on the issues in open-source projects hosted in GitHub and issue title quality is more challenging than the bug report summary quality. Chen et al. (2020) were the first to focus on the issue title quality problem in GitHub. They formulated the issue title generation into a one-sentence summarization problem. Then they proposed the method iTAPE. However, after analyzing the ROUGE-L score of the titles generated by the method iTAPE in their testing set, we find only 42.7% of issues can generate high-quality titles, whose ROUGE-L scores can exceed 0.3. Therefore, We manually analyze the main reasons for the above problem and then propose an effective quality prediction-based filter based on the DL module and the IR module, which can help to make the method iTAPE more practical.

9. Conclusion

In this study, we propose a quality-prediction-based filter `TitleGen-FL`. In particular, The DL module can automatically learn the features from a large-scale training set after analyzing the quality of the titles generated by iTAPE. The IR module can compute the IR score based on the similarity between the new issue body and the bodies in historical issues. Given the new issue, these two modules can both estimate the probability of generating a high-quality title. If either module predicts that the high-quality title cannot be generated, `TitleGen-FL` can automatically filter this issue and give a warning message. Otherwise, `TitleGen-FL` can output the title generated by the method iTAPE.

In the future, we first want to improve the issue title generation method iTAPE by improving the quality of the ground-truth title and designing more effective data preprocessing methods, which can further improve the performance of our proposed filter `TitleGen-FL`. Second, we hope our proposed filter `TitleGen-FL` can provide a useful interpretation for the filtering results. Therefore, developers can understand why these issues are filtered and take the targeted solution. Finally, we want to apply `TitleGen-FL` to the bug report summarization approaches (Jiang et al., 2017b; Liu et al., 2021) for bug tracking systems (such as Bugzilla).

CRediT authorship contribution statement

Hao Lin: Data curation, Software, Writing – original draft. **Xiang Chen:** Conceptualization, Methodology, Writing – review & editing, Supervision. **Xuejiao Chen:** Data curation, Software, Validation. **Zhanqi Cui:** Conceptualization, Methodology, Writing – review & editing. **Yun Miao:** Software, Validation. **Shan Zhou:** Writing – review & editing. **Jianmin Wang:** Writing – review & editing. **Zhan Su:** Software, Validation.

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.

Data availability

Data will be made available on request.

Acknowledgments

The authors would like to thank the editors and the anonymous reviewers for their insightful comments and suggestions, which can substantially improve the quality of this work. Hao Lin and Xiang Chen have contributed equally to this work and they are co-first authors. This work is supported in part by the National Natural Science Foundation of China (Grant nos. 61872263 and 61202006), Fund projects in the technical field of the foundation strengthening plan (Grant no. 2020-JCJQ-JJ-490), Jiangsu Provincial Frontier Leading Technology Fundamental Research Project (Grant no. BK20202001), The Nantong Application Research Plan (Grant no. JC2021124), The Open Project of Key Laboratory of Safety-Critical Software for Nanjing University of Aeronautics and Astronautics, Ministry of Industry and Information Technology, China (Grant No. NJ2020022), Student Innovation Training Program for Universities in Jiangsu Province (Grant No. 202210304112Y).

References

Ahmad, W., Chakraborty, S., Ray, B., Chang, K.-W., 2020. A transformer-based approach for source code summarization. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. pp. 4998–5007.
Alipour, A., Hindle, A., Stroulia, E., 2013. A contextual approach towards more accurate duplicate bug report detection. In: 2013 10th Working Conference on Mining Software Repositories. MSR, IEEE, pp. 183–192.
Anvik, J., Murphy, G.C., 2011. Reducing the effort of bug report triage: Recommenders for development-oriented decisions. ACM Trans. Softw. Eng. Methodol. (TOSEM) 20 (3), 1–35.
Cao, K., Chen, C., Baltes, S., Treude, C., Chen, X., 2021. Automated query reformulation for efficient search based on query logs from stack overflow. In: 2021 IEEE/ACM 43rd International Conference on Software Engineering. ICSE, IEEE, pp. 1273–1285.
Chen, S., Xie, X., Yin, B., Ji, Y., Chen, L., Xu, B., 2020. Stay professional and efficient: automatically generate titles for your bug reports. In: 2020 35th IEEE/ACM International Conference on Automated Software Engineering. ASE, IEEE, pp. 385–397.
Fleiss, J.L., 1971. Measuring nominal scale agreement among many raters. Psychol. Bull. 76 (5), 378.
Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press.
Gu, J., Lu, Z., Li, H., Li, V.O., 2016. Incorporating copying mechanism in sequence-to-sequence learning. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. Volume 1: Long Papers, pp. 1631–1640.
Haque, S., Eberhart, Z., Bansal, A., McMillan, C., 2022. Semantic similarity metrics for evaluating source code summarization. arXiv preprint arXiv:2204.01632.
Hu, X., Li, G., Xia, X., Lo, D., Jin, Z., 2018. Deep code comment generation. In: 2018 IEEE/ACM 26th International Conference on Program Comprehension. ICPC, IEEE, pp. 200–20010.
Hu, Y., Yan, M., Liu, Z., Chen, Q., Wang, B., 2021. Improving code summarization through automated quality assurance. In: 2021 IEEE 32nd International Symposium on Software Reliability Engineering. ISSRE, IEEE, pp. 486–497.
Jiang, S., Armaly, A., McMillan, C., 2017a. Automatically generating commit messages from diffs using neural machine translation. In: 2017 32nd IEEE/ACM International Conference on Automated Software Engineering. ASE, IEEE, pp. 135–146.

Jiang, H., Zhang, J., Ma, H., Nazar, N., Ren, Z., 2017b. Mining authorship characteristics in bug repositories. Sci. China Inf. Sci. 60 (1), 1–16.
Kim, Y., 2014. Convolutional neural networks for sentence classification. In: 2014 Conference on Empirical Methods in Natural Language Processing. pp. 1746–1751.
Lai, S., Xu, L., Liu, K., Zhao, J., 2015. Recurrent convolutional neural networks for text classification. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence. pp. 2267–2273.
LeClair, A., Jiang, S., McMillan, C., 2019. A neural model for generating natural language summaries of program subroutines. In: 2019 IEEE/ACM 41st International Conference on Software Engineering. ICSE, IEEE, pp. 795–806.
Li, X., Jiang, H., Liu, D., Ren, Z., Li, G., 2018. Unsupervised deep bug report summarization. In: 2018 IEEE/ACM 26th International Conference on Program Comprehension. ICPC, IEEE, pp. 144–14411.
Li, Z., Wu, Y., Peng, B., Chen, X., Sun, Z., Liu, Y., Paul, D., 2022. SeTransformer: A transformer-based code semantic parser for code comment generation. IEEE Trans. Reliab..
Li, Z., Wu, Y., Peng, B., Chen, X., Sun, Z., Liu, Y., Yu, D., 2021. SeCNN: A semantic CNN parser for code comment generation. J. Syst. Softw. 181, 111036.
Lin, C.-Y., 2004. Rouge: A package for automatic evaluation of summaries. In: Text Summarization Branches Out. pp. 74–81.
Liu, F., Li, G., Wei, B., Xia, X., Fu, Z., Jin, Z., 2020a. A self-attentional neural architecture for code completion with multi-task learning. In: Proceedings of the 28th International Conference on Program Comprehension. pp. 37–47.
Liu, P., Qiu, X., Huang, X., 2016. Recurrent neural network for text classification with multi-task learning. In: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence. pp. 2873–2879.
Liu, Z., Xia, X., Hassan, A.E., Lo, D., Xing, Z., Wang, X., 2018. Neural-machine-translation-based commit message generation: how far are we? In: Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering. pp. 373–384.
Liu, K., Yang, G., Chen, X., Yu, C., 2022. SOTitle: A transformer-based post title generation approach for stack overflow. In: Proceedings of the 29th IEEE International Conference on Software Analysis, Evolution and Reengineering. SANER 2022.
Liu, H., Yu, Y., Li, S., Geng, M., Mao, X., Liao, X., 2021. How to cherry pick the bug report for better summarization? Empir. Softw. Eng. 26 (6), 1–36.
Liu, H., Yu, Y., Li, S., Guo, Y., Wang, D., Mao, X., 2020b. Bugsum: Deep context understanding for bug report summarization. In: Proceedings of the 28th International Conference on Program Comprehension. pp. 94–105.
Lotufo, R., Malik, Z., Czarnecki, K., 2015. Modelling the 'hurried'bug report reading process to summarize bug reports. Empir. Softw. Eng. 20 (2), 516–548.
Luong, M.-T., Pham, H., Manning, C.D., 2015. Effective approaches to attention-based neural machine translation. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 1412–1421.
Mills, C., Pantiuchina, J., Parra, E., Bavota, G., Haiduc, S., 2018. Are bug reports enough for text retrieval-based bug localization? In: 2018 IEEE International Conference on Software Maintenance and Evolution. ICSME, IEEE, pp. 381–392.
Mills, C., Parra, E., Pantiuchina, J., Bavota, G., Haiduc, S., 2020. On the relationship between bug reports and queries for text retrieval-based bug localization. Empir. Softw. Eng. 25 (5), 3086–3127.
Nguyen, A.T., Nguyen, T.T., Nguyen, T.N., Lo, D., Sun, C., 2012. Duplicate bug report detection with a combination of information retrieval and topic modeling. In: 2012 Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering. IEEE, pp. 70–79.
Papineni, K., Roukos, S., Ward, T., Zhu, W.-J., 2002. Bleu: a method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics. pp. 311–318.
Rastkar, S., Murphy, G.C., Murray, G., 2014. Automatic summarization of bug reports. IEEE Trans. Softw. Eng. 40 (4), 366–380.
Saha, R.K., Lease, M., Khurshid, S., Perry, D.E., 2013. Improving bug localization using structured information retrieval. In: 2013 28th IEEE/ACM International Conference on Automated Software Engineering. ASE, IEEE, pp. 345–355.
See, A., Liu, P.J., Manning, C.D., 2017. Get to the point: Summarization with pointer-generator networks. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics. Volume 1: Long Papers, pp. 1073–1083.
Singh, R., Mangat, N.S., 2013. Elements of Survey Sampling. Vol. 15. Springer Science & Business Media.
Sutskever, I., Vinyals, O., Le, Q.V., 2014. Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems. pp. 3104–3112.
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need. In: Advances in Neural Information Processing Systems. pp. 5998–6008.
Wan, Y., Zhao, Z., Yang, M., Xu, G., Ying, H., Wu, J., Yu, P.S., 2018. Improving automatic source code summarization via deep reinforcement learning. In: Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering. pp. 397–407.
Wang, Q., Li, B., Xiao, T., Zhu, J., Li, C., Wong, D.F., Chao, L.S., 2019. Learning deep transformer models for machine translation. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. pp. 1810–1822.
Wang, B., Yan, M., Liu, Z., Xu, L., Xia, X., Zhang, X., Yang, D., 2021. Quality assurance for automated commit message generation. In: 2021 IEEE International Conference on Software Analysis, Evolution and Reengineering. SANER, IEEE, pp. 260–271.
Xuan, J., Jiang, H., Hu, Y., Ren, Z., Zou, W., Luo, Z., Wu, X., 2014. Towards effective bug triage with software data reduction techniques. IEEE Trans. Knowl. Data Eng. 27 (1), 264–280.
Yang, G., Chen, X., Cao, J., Xu, S., Cui, Z., Yu, C., Liu, K., 2021a. ComFormer: Code comment generation via transformer and fusion method-based hybrid code representation. In: 2021 8th International Conference on Dependable Systems and their Applications. DSA, IEEE, pp. 30–41.
Yang, G., Chen, X., Zhou, Y., Yu, C., 2022a. DualSC: Automatic generation and summarization of shellcode via transformer and dual learning. In: Proceedings of the 29th IEEE International Conference on Software Analysis, Evolution and Reengineering. SANER 2022.
Yang, G., Liu, K., Chen, X., Zhou, Y., Yu, C., Lin, H., 2022b. CCGIR: Information retrieval-based code comment generation method for smart contracts. Knowl.-Based Syst. 237, 107858.
Yang, G., Zhou, Y., Chen, X., Yu, C., 2021b. Fine-grained pseudo-code generation method via code feature extraction and transformer. In: 2021 28th Asia-Pacific Software Engineering Conference. APSEC, IEEE, pp. 213–222.
Zhang, T., Jiang, H., Luo, X., Chan, A.T., 2016. A literature review of research in bug resolution: Tasks, challenges and future directions. Comput. J. 59 (5), 741–773.
Zhang, J., Wang, X., Hao, D., Xie, B., Zhang, L., Mei, H., 2015. A survey on bug-report analysis. Sci. China Inf. Sci. 58 (2), 1–24.
Zhou, J., Zhang, H., Lo, D., 2012. Where should the bugs be fixed? more accurate information retrieval-based bug localization based on bug reports. In: 2012 34th International Conference on Software Engineering. ICSE, IEEE, pp. 14–24.

Hao Lin is currently pursuing the B.S. degree with School of Information Science and Technology, Nantong University. His research interests include issue title generation.

Xiang Chen received the B.Sc. degree in information management and system from Xi'an Jiaotong University, China in 2002. Then he received the M.Sc., and Ph.D. degrees in computer software and theory from Nanjing University, China in 2008 and 2011 respectively. He is with the School of Information Science and Technology at Nantong University as an associate professor. His research interests are mainly in software engineering. In particular, he is interested in empirical software engineering, mining software repositories, software maintenance and software testing. In these areas, he has published over 60 papers in refereed journals or conferences, such as IEEE Transactions on Software Engineering, Information and Software Technology, Journal of Systems and Software, IEEE Transactions on Reliability, Journal of Software: Evolution and Process, Software Quality Journal, Journal of Computer Science and Technology, ASE, ICSME, SANER and COMPSAC. He is a senior member of CCF, China, and a member of IEEE and ACM.

Xuejiao Chen is currently pursuing the B.S. degree with School of Information Science and Technology, Nantong University. Her research interests include issue title generation.

Zhanqi Cui is with computer school at Beijing Information Science and Technology University as an associate professor. His research interests are mainly in software engineering.

Yun Miao is currently pursuing the B.S. degree with School of Information Science and Technology, Nantong University. Her research interests include issue title generation.

Shan Zhou is with the technology and engineering center for space utilization, Chinese Academy of Sciences as an engineer. Her research interests are mainly in software engineering.

Jianmin Wang is with the technology and engineering center for space utilization, Chinese Academy of Sciences as an assistant researcher. His research interests are mainly in software engineering.

Zhan Su is currently pursuing the B.S. degree with School of Information Science and Technology, Nantong University. His research interests include issue title generation.