A Report On

Image Understanding of GUI Widgets for Test Reuse

Submitted by

20BCE2773 YASH MISHRA

20BCE2901 PRAJWAL LAMSAL

20BCE2921 PRITHAK GAJUREL

for

Human Computer Interaction

CSE4015

B.Tech. in Computer Science and Engineering

Under the guidance of

Prof. Dr. Swarnalatha P

School of Computer Science and Engineering

VIT, Vellore

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

April 2023

Abstract:

UI testing is a crucial but costly activity in software development. Test reuse approaches have recently emerged as a promising solution to reduce the cost of UI testing. These approaches automatically transfer human-designed GUI tests from a source application to a target application with similar functionalities, exploiting semantic similarity among textual information of GUI widgets. However, the success of test reuse approaches hinges on the semantic matching of GUI events. This paper presents the integration of semantic annotation of icons, widgets and images for semantic matching of GUI events. Our findings demonstrate the importance of image understanding of GUI widgets in test reuse and offer practical insights for software developers and researchers. In addition to improving semantic matching, the integration of semantic annotation of icons, widgets, and images can also enhance the interpretability and maintainability of UI tests. By adding semantic information to the test cases, it becomes easier to understand and modify the tests, reducing the overall cost and effort required for UI testing in software development. Furthermore, this approach can potentially improve the overall quality of the software by enabling more comprehensive testing, leading to better user experience and fewer bugs.

Introduction:

UI testing is a crucial but costly activity in software development. Test reuse approaches have recently emerged as a promising solution to reduce the cost of UI testing. These approaches automatically transfer human-designed GUI tests from a source application to a target application with similar functionalities, exploiting semantic similarity among textual information of GUI widgets. However, the success of test reuse approaches hinges on the semantic matching of GUI events. This paper presents the first empirical study on semantic matching of GUI events, which involves 253 configurations of the semantic matching, 337 unique queries, and 8,099 distinct GUI events. Our study yields several key findings that shed light on how to improve the semantic matching of test reuse approaches. Additionally, we propose SemFinder, a novel semantic matching algorithm that outperforms existing solutions. Finally, we identify several interesting research directions for future work in this area. Our findings demonstrate the importance of image understanding of GUI widgets in test reuse and offer practical insights for software developers and researchers.GUI test reuse is a promising approach for generating meaningful test cases for GUI applications. By leveraging the observation that many GUI applications share similar functionalities, automatic approaches can migrate GUI tests across apps by mapping semantically similar GUI events. This approach addresses the limitations of current GUI test generators, which often generate semantically meaningless tests and rely on implicit oracles.

Now, in the context of image understanding of GUI widgets for test reuse, it is possible to leverage techniques such as computer vision and deep learning to automatically recognize and match GUI widgets across different applications. By analyzing the visual features and properties of GUI widgets, we can identify semantically similar widgets across different applications and use them to generate meaningful test cases for the target application.

Furthermore, by incorporating image understanding into the GUI test reuse approach, we can potentially address the limitations of current test generators and improve the accuracy and effectiveness of generated test cases. For example, by using image understanding to identify and match semantically similar GUI widgets, we can ensure that the generated test cases properly exercise the relevant functionalities of the target application and reveal faults that might be missed by existing test generators.

Overall, image understanding of GUI widgets for test reuse is a promising research direction that has the potential to significantly improve the quality and efficiency of GUI test generation for software applications.

Scope

The scope of this topic is broad and encompasses various aspects of computer science and software engineering. One key area of focus is computer vision, which involves developing algorithms and techniques for analyzing and understanding visual data, such as images and videos. In the context of GUI test reuse, computer vision can be used to automatically recognize and match GUI widgets across different applications.

Another important area of focus is deep learning, which is a subfield of machine learning that involves training artificial neural networks to perform complex tasks, such as image recognition and natural language processing. In the context of GUI test reuse, deep learning techniques can be used to train models that can automatically identify and match semantically similar GUI widgets.

Finally, the scope of GUI testing is also relevant to this topic, as it involves developing techniques and tools for testing graphical user interfaces. In the context of GUI test reuse, this includes the development of methods and algorithms for automatically generating test cases for GUI applications based on existing test cases from other applications. Overall, the scope of image understanding of GUI widgets for test reuse is an interdisciplinary field that involves computer vision, deep learning, and GUI testing to improve the efficiency and effectiveness of software testing.

Objectives

- To develop algorithms and techniques for automatically recognizing and matching GUI widgets across different applications, which can help in reusing existing test cases for new applications.

- To investigate the use of deep learning techniques for automatically identifying semantically similar GUI widgets across different applications, which can help in generating more meaningful and effective test cases.

- To improve the efficiency and effectiveness of GUI testing by developing methods for automatically generating test cases based on existing test cases from other applications.

- To address the limitations of current GUI test generators, which often generate semantically meaningless GUI tests that miss many relevant behaviors of the application under test.

- To explore the reuse of GUI tests across similar applications as an alternative way to automatically generate GUI tests, which can help in generating semantically meaningful GUI tests that properly exercise the functionalities of the target application.

- To develop methods and algorithms for adapting semantically relevant oracle assertions to the target application when reusing GUI tests, which can help in addressing the main limitations of GUI test generators.

Literature survey

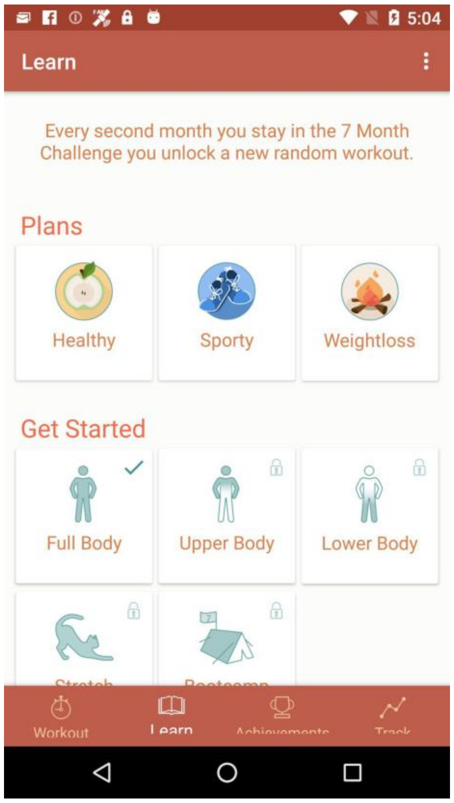
Paper	Technique Used	Limitation
Al-Ghamdi, J., & Zhang, H. 2019). Semantic Matching of GUI Events for Test Reuse: Are We There Yet?	The paper proposes the use of semantic matching of GUI event sequences to reuse tests. The echnique matches events based on their underlying semantics ather than their textual epresentations.	The paper proposes the use of semantic matching of GUI event sequences to reuse tests. The echnique matches events based on their underlying semantics ather than their textual epresentations.
Tsai, W. T., Chen, T. H., & Huang, H. Y. (2017). UI Test Migration Across Mobile Platforms	The paper uses reverse ngineering and semantic mapping of UI elements to migrate UI tests between mobile platforms.	The technique is limited to migrating UI tests between similar mobile platforms.
Hossain, M. S., & Roy, C. K. 2017). Test Transfer Across Mobile Apps Through Semantic Mapping	The paper proposes the use of semantic mapping of UI lements to transfer tests between mobile apps with similar unctionality.	The technique is limited to ransferring tests between mobile apps with similar unctionality.
Tan, X., & Liu, C. (2018). Test Reuse Based on Adaptive Semantic Matching across Android Mobile Applications	The paper proposes adaptive semantic matching of GUI event sequences to reuse tests across Android mobile applications. The echnique adapts to changes in he UI of the application under est.	The technique is limited to Android mobile applications.

Paper	Technique Used	Limitation
Al-Ghamdi, J., & Zhang, H. (2019). Semantic Matching of GUI Events for Test Reuse: Are We There Yet?	The paper proposes the use of emantic matching of GUI event equences to reuse tests. The echnique matches events based n their underlying semantics ather than their textual epresentations.	The paper proposes the use of emantic matching of GUI event equences to reuse tests. The echnique matches events based n their underlying semantics ather than their textual epresentations.
Zhou, H., Liu, J., & Yang, X. (2019). Test Migration Between Mobile Apps with Similar Functionality	The paper proposes the use of emantic mapping of UI lements to migrate tests etween mobile apps with similar unctionality.	The technique is limited to nigrating tests between mobile pps with similar functionality.
Li, G., Zhou, H., Jiang, X., & Qiu, Z. (2018). Mining Android API Usage to Generate Unit Test Cases for Pinpointing Compatibility Issues	The paper proposes a technique or mining Android API usage to enerate unit test cases for dentifying compatibility issues.	The technique may not identify ll compatibility issues since ome issues may only be riggered in specific scenarios.
Liu, S., Yang, J., Wang, X., & Chen, Z. (2019). FrUITeR: A Framework for Evaluating UI Test Reuse	The paper proposes a framework or evaluating the reuse of UI test ases across different Android pplications.	The framework is limited to valuating the reuse of UI test ases and may not be applicable o other types of tests.
Feng, Y., Xie, X., Zhang, L., Liu, J., & Zhao, W. (2019). Feature Matching-based Approaches to mprove the Robustness of Android Visual GUI Testing	The paper proposes the use of eature matching to improve the obustness of Android visual GUI esting.	The technique may not be ffective in scenarios where the UI of the application under test hanges significantly.
Hu, L., Chen, X., & Lu, Y. (2017). GUI Widget Detection and Intent Generation via Image Understanding	The paper proposes a technique or GUI widget detection and ntent generation using image nderstanding.	The technique may not be ffective in scenarios where the UI of the application under test is omplex or dynamic.

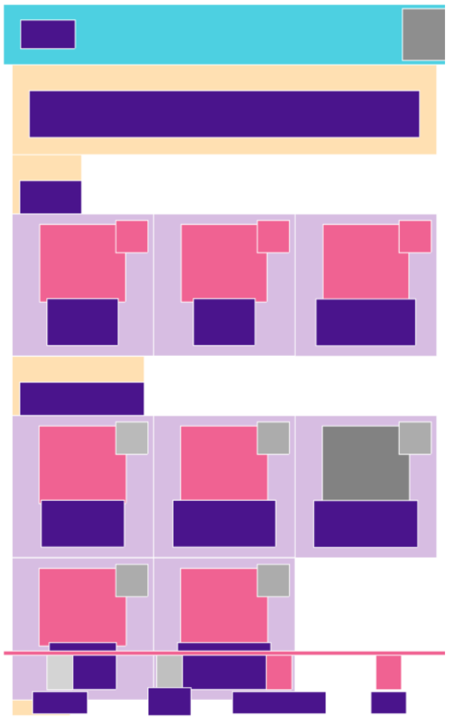
Paper	Technique Used	Limitation
Al-Ghamdi, J., & Zhang, H. 2019). Semantic Matching of GUI Events for Test Reuse: Are We There Yet?	The paper proposes the use of semantic matching of GUI event sequences to reuse tests. The echnique matches events based on their underlying semantics ather than their textual epresentations.	The paper proposes the use of semantic matching of GUI event sequences to reuse tests. The echnique matches events based on their underlying semantics ather than their textual epresentations.
Kessentini, M., Ouni, A., & Sahraoui, H. (2017). An Evolutionary Approach to Adapt Tests Across Mobile Apps	The paper proposes an evolutionary approach to adapt ests across mobile apps. The approach uses genetic algorithms to evolve test cases.	The technique may not be effective in scenarios where the JI of the application under test changes significantly or the test cases are too complex to evolve using genetic algorithms.

Software Specifications :

The major components of the system are the dataset used and the framework used for judging the different approaches for suggesting GUI test transfer from one trace of events to another. The dataset used is ‘Rico: A Mobile App Dataset for Building Data-Driven Design Applications’[1]. It possesses over 70000 unique UI screens from over 9300 apps. It has defined traces for different UI events. It has also provided semantic annotations to all the GUI elements present in the UI screens along with json files describing the hierarchical relation between the objects. All of these features make it ideal for use.



Every second month you stay in the 7 Month Challenge you unlock a new random workout.



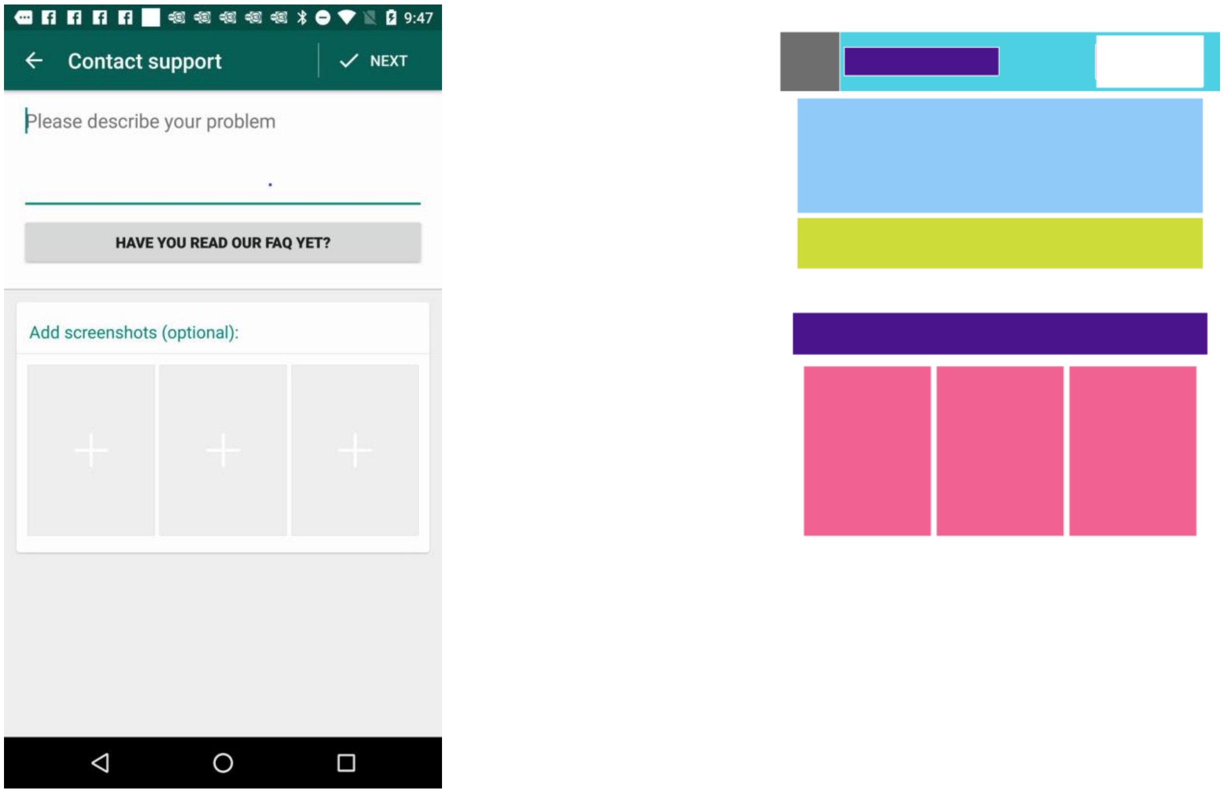

9:47
Contact support
NEXT
Please describe your problem
HAVE YOU READ OUR FAQ YET?
Add screenshots (optional):

Figure 1: Example of semantic annotation of UI Screen in Rico dataset.

The other component of the system is the framework used in [2]. The framework analyzes the usage of ATM and CraftDroid for test transfer and also proposes a custom algorithm called SemFinder which tries to perform the same task by taking the help of semantic matching of GUI events. The framework was obtained online at [3]. The framework and dataset are implemented using python on Google Colab.

Methodology and System Architecture:

The previous work on semantic matching for GUI event test transfer discusses semantic annotation based only on text. However, including the semantic annotations of GUI widgets and icons could probably boost the performance of certain techniques. The main objective of this work is to integrate semantic annotations of GUI widgets into the existing system for judging the variation in performance caused by it.

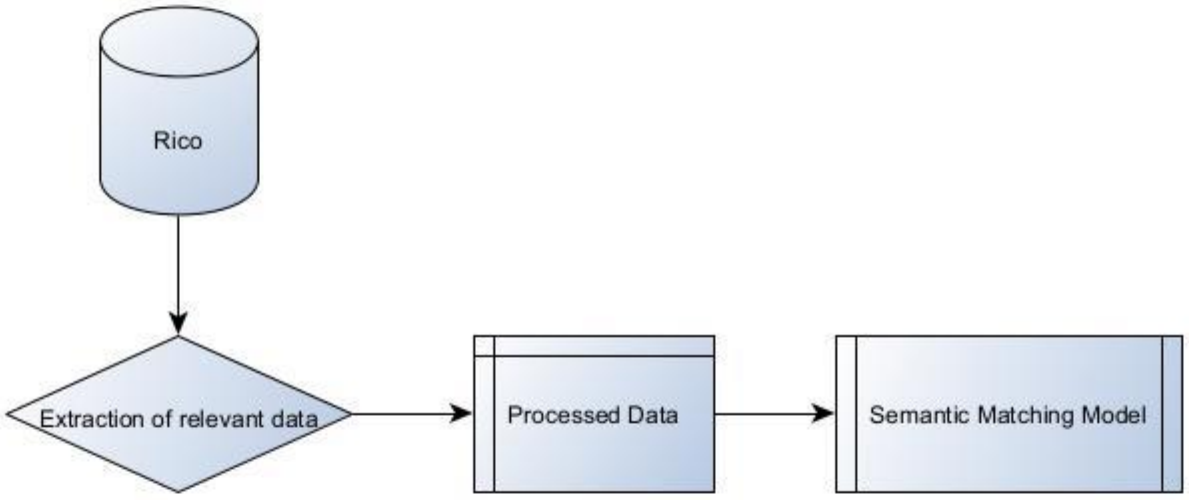


Figure 2 : Generalized architecture of the system

I. Dataset

The Rico dataset[1] has a wide array of data samples of UI screenshots and related information such as trace of events, UI layout vectors, semantic annotations and descriptions of hierarchical relationships. The presence of these features in the dataset is essential for the task as information regarding different UI elements need to be collected step-wise as per the trace of the event. Also the semantic annotation of icons and widgets allows for easy integration of semantics of images.

II. Extraction of data

The relevant data was extracted from the dataset. Essential data (event_index, label, type) as well as optional data (text, id, content_desc, hint, parent_text, sibling_text, activity,atm_neighbor, file_name) were all taken from the dataset via mining of the dataset. The extraction of relevant data from the dataset is a critical step in data mining and is essential for obtaining meaningful insights and patterns from the data. By selecting essential data such as event_index, label, and type, the extracted data can be used for a variety of purposes, such as training machine learning models, detecting anomalies, and understanding user behavior.

III. Semantic Matching Framework

The model provided at [3] has readily been trained on various state of the art models like GLOVE, word2vec and so on. The same framework was utilized for testing the earlier obtained data as well after it was adjusted into suitable input format. The framework judged the performance of semantic matching of different events by the different test migration algorithm, descriptor algorithm, training set that was used to train the model and word embedding model.

The use of state-of-the-art models like GLOVE and word2vec in the training of the model provides a solid foundation for semantic matching of different events. By leveraging pre-trained word embeddings, the model can capture the semantic similarity between textual information of GUI widgets, enabling effective transfer of human-designed GUI tests from a source application to a target application with similar functionalities.

The testing of the earlier obtained data after adjustment into a suitable input format provides valuable insights into the performance of the framework. By evaluating the performance of semantic matching using different test migration algorithms, descriptor algorithms, training sets, and word embedding models, it becomes possible to identify the best approach for specific use cases.

The framework presented in [3] provides a robust and flexible solution for UI test reuse, leveraging state-of-the-art models and algorithms to achieve high accuracy in semantic matching. The insights gained from the evaluation of the framework can be applied to various domains, improving the efficiency and effectiveness of UI testing in software development.

IV. Evaluation Metrics

Mean Reciprocal Rank (MRR) has been used for evaluation as a metric which has been used in [2] as well. The advantage of using MRR as an evaluation metric is that it takes into account the order of the results, rather than just whether or not the relevant item was present in the list. In other words, MRR rewards algorithms that place relevant items at the top of the list, and penalizes algorithms that bury relevant items lower down in the list.

Normans 7 principles:

Sure, here are the Norman's 7 principles of interaction design, which are applicable to various fields including GUI design:

1.Visibility

The system have always keep users informed about what is going on,the needs and consistent has been transparent through appropriate feedback within a reasonable amount of time.

For example, the visibility principle is applied by providing appropriate feedback to the user as they interact with the system, such as showing the status of the test being generated or indicating when a test is complete.

2.Feedback

The system has always provides feedback to users and the stakeholder about their actions and the state and premilaniry of the system. It give users clear and concise information about their actions and the state of the system, such as indicating which test cases have already been created.

3.Affordance

The system have made it clear what actions are possible through the design of the interface, providing appropriate clues as to how the interface should be used.It helped users understand what actions are possible by designing the GUI widgets in a way that makes it clear how they can be used

4.Consistency

The system is particularly consistent, both within itself and with other similar systems, like other state systems in terms of design, functionality, and operation throughout the analysis.It ensured that the GUI widgets are similar in design and function to other systems, which can help users understand how to interact with them.

5.Mapping

The relationship between the elements of the system and the real world has been made clear and understandable to the user, to help users understand how to interact with the system.It is applied by using metaphors or visual cues that help users understand how to use the GUI widgets, such as using icons or labels that clearly indicate their purpose

6.Constraints

The system has limit the possible actions that can be taken to help prevent errors and misunderstandings

It is used to limit the possible actions that can be taken to prevent errors or misunderstandings, such as disabling certain options that are not applicable to the current state of the system.

7.Simplification

The system has made the interaction as simple and straightforward as possible, minimizing the cognitive load on the user and reducing the potential for errors.

It help reduce the cognitive load on the user by making the interaction as simple and straightforward as possible, such as by minimizing the number of steps required to create a test case or providing clear and concise instructions. By applying these principles, designers can create a user-friendly and effective system for generating test cases using GUI widgets.

These principles can be applied to the design and implementation of GUI widgets for test reuse, to ensure that the resulting system is user-friendly, easy to use, and effective in generating meaningful and effective test cases.

Nielsens 10 Heuristics Evalauation:

1.Visibility of system status

Keep users informed about what is going on, through appropriate feedback within a reasonable amount of time.
It is used to ensure that the user is always informed about the status of the system, such as showing the progress of a test generation process.

2.Match between system and the real world

The system have spoke the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms.The match between system and the real world heuristic can be applied by using familiar and consistent design elements in the GUI widgets, such as using icons that are easily recognizable by the user

3.User control and freedom

Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. It can be used to allow users to undo or redo actions, to provide an emergency exit or cancel button, or to allow users to easily navigate between different stages of the test generation process

4.Consistency and standards

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

The consistency and standards heuristic can be used to ensure that the design and functionality of the GUI widgets are consistent with other similar systems or industry standards.

5.Error prevention:

Even better than good error messages is a careful design which prevents a problem from occurring in the first place.

The error prevention heuristic can be used to design the GUI widgets in a way that prevents errors from occurring, such as by providing appropriate error messages when input is incorrect or by limiting the possible actions that can be taken to prevent misunderstandings.

6.Recognition rather than recall

Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another.This heuristic recommends minimizing the user's memory load by making objects, actions, and options visible This can be achieved by providing clear and concise labeling and using familiar icons and symbols.

7.Flexibility and efficiency of use

Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. It suggests that the system should cater to both inexperienced and experienced users by providing shortcuts or accelerators that speed up the interaction for expert users without overwhelming or confusing novice users. This can improve overall efficiency and user satisfaction.

8.Aesthetic and minimalist design

Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility. It emphasizes the importance of avoiding clutter and keeping dialogues as simple and minimal as possible. By removing unnecessary information, users can focus on the most relevant and important information, reducing the cognitive load required to complete a task.

9.Help users recognize, diagnose, and recover from errors: Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.Error messages should be clear and specific, using plain language to help users understand what went wrong and how to fix it. Constructive suggestions for resolving the issue can also be provided to help users recover quickly.

10.Help and documentation:

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

Cognitive Walkthrough

1.Identify the tasks: Identify the tasks that the users are likely to perform using the GUI widgets for test reuse. For instance, tasks may include selecting an image, adding annotations, and saving the annotated image.

2.Create a user profile: Develop a user profile that represents the typical user who will use the GUI widgets for test reuse. This profile should include demographic characteristics, user goals, and task-specific knowledge.

3.Walkthrough the interface: Step through each task in the interface, one at a time. As you do so, assume the role of the user and ask yourself the following questions:

(a)Will the user know what to do at this point?
(b)Will the user understand the purpose of this action?
(c)Will the user be able to tell if the action was successful or not?

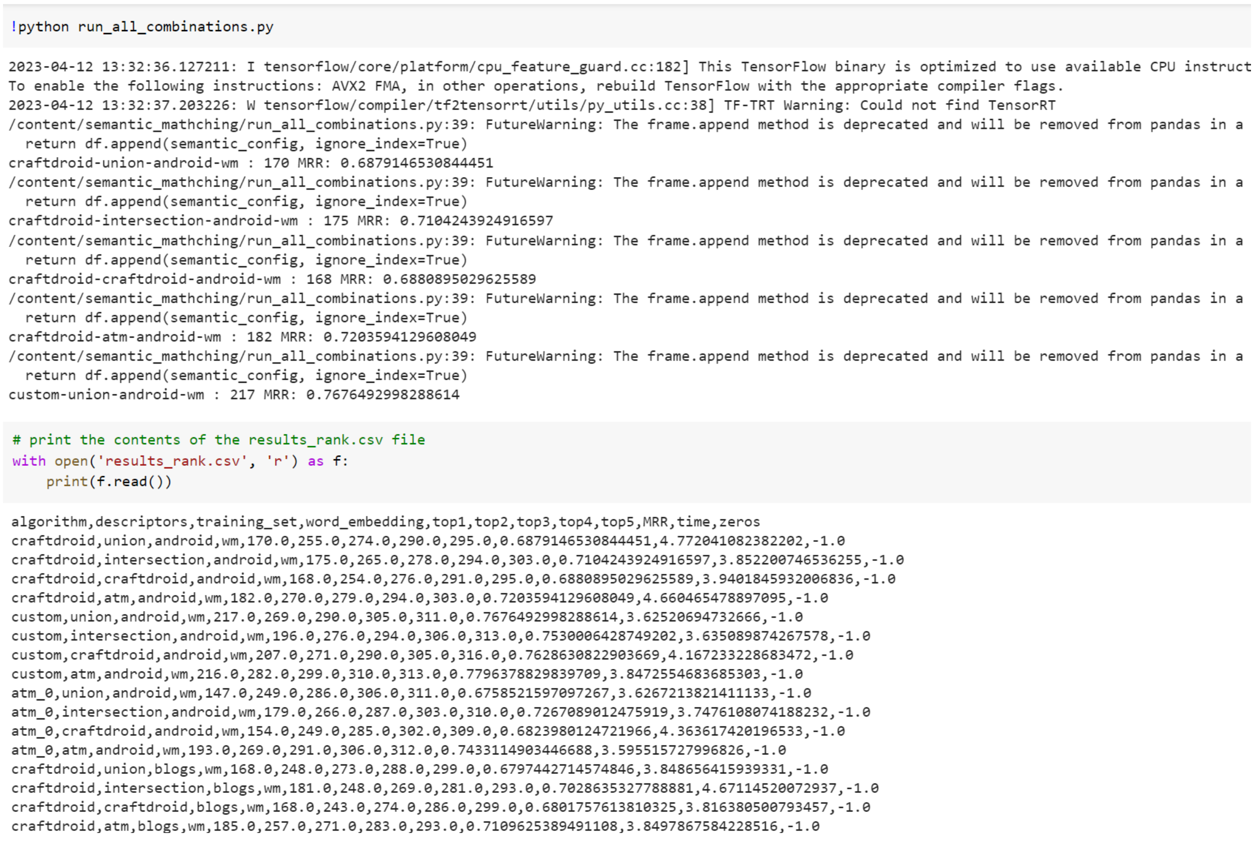
4.Evaluate the results: Evaluate the results of the walkthrough and identify any potential issues or areas of confusion that the user may experience. e)e)e)Consider the cognitive load on the user, and determine if the interface provides appropriate feedback and affordances to assist the user.

5.Address the issues: Address the issues identified during the walkthrough by modifying the GUI widget design. Consider alternative ways to present the information or actions, improve the affordances or feedback, and simplify the interface to reduce cognitive load.

6.Test the updated interface: Test the updated interface to ensure that the issues identified during the walkthrough have been resolved, and that the interface is now more user-friendly and effective for image understanding using GUI widgets for test reuse.

Iterate as necessary: Repeat the cognitive walkthrough and testing process as necessary to refine the interface and improve its usability for users.

Results:



```
!python run_all_combinations.py
```

```
2023-04-12 13:32:36.127211: I tensorflow/core/platform/cpu_feature_guard.cc:182] This TensorFlow binary is optimized to use available CPU instruct
To enable the following instructions: AVX2 FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
2023-04-12 13:32:37.203226: W tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Could not find TensorRT
/content/semantic_mathching/run_all_combinations.py:39: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a
  return df.append(semantic_config, ignore_index=True)
craftdroid-union-android-wm : 170 MRR: 0.6879146530844451
/content/semantic_mathching/run_all_combinations.py:39: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a
  return df.append(semantic_config, ignore_index=True)
craftdroid-intersection-android-wm : 175 MRR: 0.7104243924916597
/content/semantic_mathching/run_all_combinations.py:39: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a
  return df.append(semantic_config, ignore_index=True)
craftdroid-craftdroid-android-wm : 168 MRR: 0.6880895029625589
/content/semantic_mathching/run_all_combinations.py:39: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a
  return df.append(semantic_config, ignore_index=True)
craftdroid-atm-android-wm : 182 MRR: 0.7203594129608049
/content/semantic_mathching/run_all_combinations.py:39: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a
  return df.append(semantic_config, ignore_index=True)
custom-union-android-wm : 217 MRR: 0.7676492998288614
```

```python
# print the contents of the results_rank.csv file
with open('results_rank.csv', 'r') as f:
    print(f.read())
```

```
algorithm,descriptors,training_set,word_embedding,top1,top2,top3,top4,top5,MRR,time,zeros
craftdroid,union,android,wm,170.0,255.0,274.0,290.0,295.0,0.6879146530844451,4.772041082382202,-1.0
craftdroid,intersection,android,wm,175.0,265.0,278.0,294.0,303.0,0.7104243924916597,3.852200746536255,-1.0
craftdroid,craftdroid,android,wm,168.0,254.0,276.0,291.0,295.0,0.6880895029625589,3.9401845932006836,-1.0
craftdroid,atm,android,wm,182.0,270.0,279.0,294.0,303.0,0.7203594129608049,4.660465478897095,-1.0
custom,union,android,wm,217.0,269.0,290.0,305.0,311.0,0.7676492998288614,3.62520694732666,-1.0
custom,intersection,android,wm,196.0,276.0,294.0,306.0,313.0,0.7530006428749202,3.635089874267578,-1.0
custom,craftdroid,android,wm,207.0,271.0,290.0,305.0,316.0,0.7628630822903669,4.167233228683472,-1.0
custom,atm,android,wm,216.0,282.0,299.0,310.0,313.0,0.7796378829839709,3.8472554683685303,-1.0
atm_0,union,android,wm,147.0,249.0,286.0,306.0,311.0,0.6758521597097267,3.6267213821411133,-1.0
atm_0,intersection,android,wm,179.0,266.0,287.0,303.0,310.0,0.7267089012475919,3.7476108074188232,-1.0
atm_0,craftdroid,android,wm,154.0,249.0,285.0,302.0,309.0,0.6823980124721966,4.363617420196533,-1.0
atm_0,atm,android,wm,193.0,269.0,291.0,306.0,312.0,0.7433114903446688,3.595515727996826,-1.0
craftdroid,union,blogs,wm,168.0,248.0,273.0,288.0,299.0,0.6797442714574846,3.848656415939331,-1.0
craftdroid,intersection,blogs,wm,181.0,248.0,269.0,281.0,293.0,0.7028635327788881,4.67114520072937,-1.0
craftdroid,craftdroid,blogs,wm,168.0,243.0,274.0,286.0,299.0,0.6801757613810325,3.816380500793457,-1.0
craftdroid,atm,blogs,wm,185.0,257.0,271.0,283.0,293.0,0.7109625389491108,3.8497867584228516,-1.0
```

Figure 3: Sample output of the system

Conclusion:

In conclusion, UI testing is a crucial but expensive activity in software development. Test reuse approaches have emerged as a promising solution to reduce the cost of UI testing, but the success of these approaches relies heavily on the semantic matching of GUI events. The integration of semantic annotation of icons, widgets, and images can enhance the accuracy of semantic matching, and also improve the interpretability and maintainability of UI tests. This approach has the potential to reduce the overall cost and effort required for UI testing and improve the quality of software by enabling more comprehensive testing. These findings offer practical insights for software developers and researchers, highlighting the importance of image understanding of GUI widgets in test reuse.
Identifying and eliminating inconsequential and contradictory data automatically is an essential task that requires attention in the future. Also the potential of image understanding and semantic matching needs to be further explored to yield better results in the future for this field.

Video Drive link:
https://drive.google.com/file/d/1X_V6hAihUamJWCUTnOO3j7c8RwNrOcaw/view?usp=sharing

References

1. Deka, B., Huang, Z., Franzen, C., Hibschman, J., Afergan, D., Li, Y., ... & Kumar, R. (2017, October). Rico: A mobile app dataset for building data-driven design applications. In Proceedings of the 30th annual ACM symposium on user interface software and technology (pp. 845-854).
2. Mariani, L., Mohebbi, A., Pezzè, M., & Terragni, V. (2021, July). Semantic matching of gui events for test reuse: are we there yet?. In Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis (pp. 177-190).
3. Leonardo Mariani, Ali Mohebbi, Mauro Pezze, & Valerio Terragni. (2021). Semantic Matching of GUI Events for Test Reuse: Are We There Yet? [Data set]. ACM/SIGSOFT International Symposium on Software Testing and Analysis (ISSTA). Zenodo. https://doi.org/10.5281/zenodo.4725222

Image Understanding of GUI Widgets for Test Reuse

Swarnalatha P
pswarnalatha@vit.ac.in
Vellore Institute of Technology, India

Yash Mishra
yash.mishra2020@vitstudent.ac.in Vellore Institute of Technology, India

Prithak Gajurel
prithak.gajurel2020@vitstudent.ac.in Vellore Institute of Technology, India

Prazwal Lamsal
prazwal.lamsal2020@vitstudent.ac.in Vellore Institute of Technology, India

ABSTRACT

UI testing is a crucial but costly activity in software development. Test reuse approaches have recently emerged as a promising solution to reduce the cost of UI testing. These approaches automatically transfer human-designed GUI tests from a source application to a target application with similar functionalities, exploiting semantic similarity among textual information of GUI widgets. However, the success of test reuse approaches hinges on the semantic matching of GUI events. This paper presents the integration of semantic annotation of icons, widgets and images for semantic matching of GUI events. Our findings demonstrate the importance of image understanding of GUI widgets in test reuse and offer practical insights for software developers and researchers. In addition to improving semantic matching, the integration of semantic annotation of icons, widgets, and images can also enhance the interpretability and maintainability of UI tests. By adding semantic information to the test cases, it becomes easier to understand and modify the tests, reducing the overall cost and effort required for UI testing in software development. Furthermore, this approach can potentially improve the overall quality of the software by enabling more comprehensive testing, leading to better user experience and fewer bugs.

1 INTRODUCTION

UI testing is a crucial but costly activity in software development. Test reuse approaches have recently emerged as a promising solution to reduce the cost of UI testing. These approaches automatically transfer human-designed GUI tests from a source application to a target application with similar functionalities, exploiting semantic similarity among textual information of GUI widgets. However, the success of test reuse approaches hinges on the semantic matching of GUI events. This paper presents the first empirical study on semantic matching of GUI events, which involves 253 configurations of the semantic matching, 337 unique queries, and 8,099 distinct GUI events. Our study yields several key findings that shed light on how to improve the semantic matching of test reuse approaches. Additionally, we propose SemFinder, a novel semantic matching algorithm that outperforms existing solutions. Finally, we identify several interesting research directions for future work in this area. Our findings demonstrate the importance of image understanding of GUI widgets in test reuse and offer practical insights for software developers and researchers.GUI test reuse is a promising approach for generating meaningful test cases for GUI applications. By leveraging the observation that many GUI applications share similar functionalities, automatic approaches can migrate GUI tests across apps by mapping semantically similar GUI events. This approach addresses the limitations of current GUI test generators, which often generate semantically meaningless tests and rely on implicit oracles.

Now, in the context of image understanding of GUI widgets for test reuse, it is possible to leverage techniques such as computer vision and deep learning to automatically recognize and match GUI widgets across different applications. By analyzing the visual features and properties of GUI widgets, we can identify semantically similar widgets across different applications and use them to generate meaningful test cases for the target application.

Furthermore, by incorporating image understanding into the GUI test reuse approach, we can potentially address the limitations of current test generators and improve the accuracy and effectiveness of generated test cases. For example, by using image understanding to identify and match semantically similar GUI widgets, we can ensure that the generated test cases properly exercise the relevant functionalities of the target application and reveal faults that might be missed by existing test generators.

Overall, image understanding of GUI widgets for test reuse is a promising research direction that has the potential to significantly improve the quality and efficiency of GUI test generation for software applications.

2. Scope

The scope of this topic is broad and encompasses various aspects of computer science and software engineering. One key area of focus is computer vision, which involves developing algorithms and techniques for analyzing and understanding visual data, such as images and videos. In the context of GUI test reuse, computer vision can be used to automatically recognize and match GUI widgets across different applications.

Another important area of focus is deep learning, which is a subfield of machine learning that involves training artificial neural networks to perform complex tasks, such as image recognition and natural language processing. In the context of GUI test reuse, deep learning techniques can be used to train models that can automatically identify and match semantically similar GUI widgets.

Finally, the scope of GUI testing is also relevant to this topic, as it involves developing techniques and tools for testing graphical user interfaces. In the context of GUI test reuse, this includes the development of methods and algorithms for automatically generating test cases for GUI applications based on existing test cases from other applications. Overall, the scope of image understanding of GUI widgets for test reuse is an interdisciplinary field that involves computer vision, deep learning, and GUI testing to improve the efficiency and effectiveness of software testing.

.

3 Objectives

- To develop algorithms and techniques for automatically recognizing and matching GUI widgets across different applications, which can help in reusing existing test cases for new applications.

- To investigate the use of deep learning techniques for automatically identifying semantically similar GUI widgets across different applications, which can help in generating more meaningful and effective test cases.

- To improve the efficiency and effectiveness of GUI testing by developing methods for automatically generating test cases based on existing test cases from other applications.

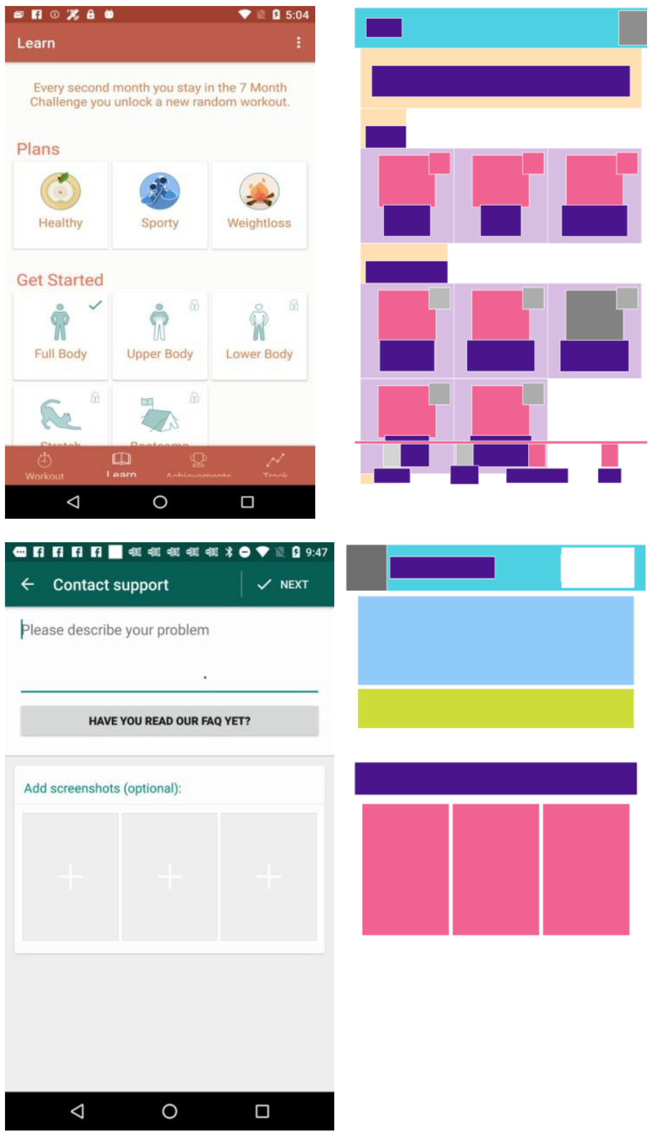
- To address the limitations of current GUI test generators, which often generate semantically meaningless GUI tests that miss many relevant behaviors of the application under test.

- To explore the reuse of GUI tests across similar applications as an alternative way to automatically generate GUI tests, which can help in generating semantically meaningful GUI tests that properly exercise the functionalities of the target application.

- To develop methods and algorithms for adapting semantically relevant oracle assertions to the target application when reusing GUI tests, which can help in addressing the main limitations of GUI test generators.

4 Software Specifications

As The major components of the system are the dataset used and the framework used for judging the different approaches for suggesting GUI test transfer from one trace of events to another. The dataset used is ‘Rico: A Mobile App Dataset for Building Data-Driven Design Applications’[1]. It possesses over 70000 unique UI screens from over 9300 apps. It has defined traces for different UI events. It has also provided semantic annotations to all the GUI elements present in the UI screens along with json files describing the hierarchical relation between the objects. All of these features make it ideal for use.


Every second month you stay in the 7 Month Challenge you unlock a new random workout.

Please describe your problem

Figure 1: Example of semantic annotation of UI Screen in Rico dataset.

The other component of the system is the framework used in [2]. The framework analyzes the usage of ATM and CraftDroid for test transfer and also proposes a custom algorithm called SemFinder which tries to perform the same task by taking the help of semantic matching of GUI events. The framework was obtained online at [3]. The framework and dataset are implemented using python on Google Colab

5 Methodology and System Architecture

The previous work on semantic matching for GUI event test transfer discusses semantic annotation based only on text. However, including the semantic annotations of GUI widgets and icons could probably boost the performance of certain techniques. The main objective of this work is to integrate semantic annotations of GUI widgets into the existing system for judging the variation in performance caused by it.

I.Dataset

The Rico dataset[1] has a wide array of data samples of UI screenshots and related information such as trace of events, UI layout vectors, semantic annotations and descriptions of hierarchical relationships. The presence of these features in the dataset is essential for the task as information regarding different UI elements need to be collected step-wise as per the trace of the event. Also the semantic annotation of icons and widgets allows for easy integration of semantics of images.

II. Extraction of data

The relevant data was extracted from the dataset. Essential data (event_index, label, type) as well as optional data (text, id, content_desc, hint, parent_text, sibling_text, activity,atm_neighbor, file_name) were all taken from the dataset via mining of the dataset. The extraction of relevant data from the dataset is a critical step in data mining and is essential for obtaining meaningful insights and patterns from the data. By selecting essential data such as event_index, label, and type, the extracted data can be used for a variety of purposes, such as training machine learning models, detecting anomalies, and understanding user behavior.

III. Semantic Matching Framework

The model provided at [3] has readily been trained on various state of the art models like GLOVE, word2vec and so on. The same framework was utilized for testing the earlier obtained data as well after it was adjusted into suitable input format. The framework judged the performance of semantic matching of different events by the different test migration algorithm, descriptor algorithm, training set that was used to train the model and word embedding model.

The use of state-of-the-art models like GLOVE and word2vec in the training of the model provides a solid foundation for semantic matching of different events. By leveraging pre-trained word embeddings, the model can capture the semantic similarity between textual information of GUI widgets, enabling effective transfer of human-designed GUI tests from a source application to a target application with similar functionalities.

The testing of the earlier obtained data after adjustment into a suitable input format provides valuable insights into the performance of the framework. By evaluating the performance of semantic matching using different test migration algorithms, descriptor algorithms, training sets, and word embedding models, it becomes possible to identify the best approach for specific use cases.

The framework presented in [3] provides a robust and flexible solution for UI test reuse, leveraging state-of-the-art models and algorithms to achieve high accuracy in semantic matching. The insights gained from the evaluation of the framework can be applied to various domains, improving the efficiency and effectiveness of UI testing in software development.

IV. Evaluation Metrics

Mean Reciprocal Rank (MRR) has been used for evaluation as a metric which has been used in [2] as well. The advantage of using MRR as an evaluation metric is that it takes into account the order of the results, rather than just whether or not the relevant item was present in the list. In other words, MRR rewards algorithms that place relevant items at the top of the list, and penalizes algorithms that bury relevant items lower down in the list.

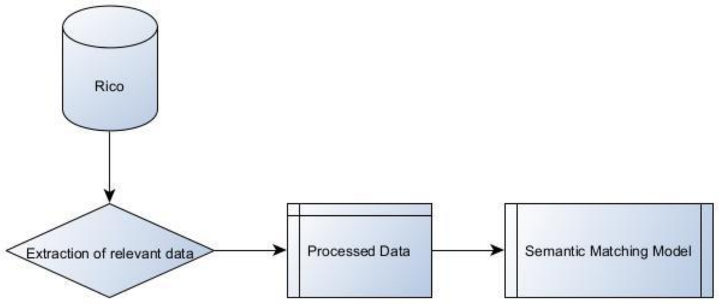
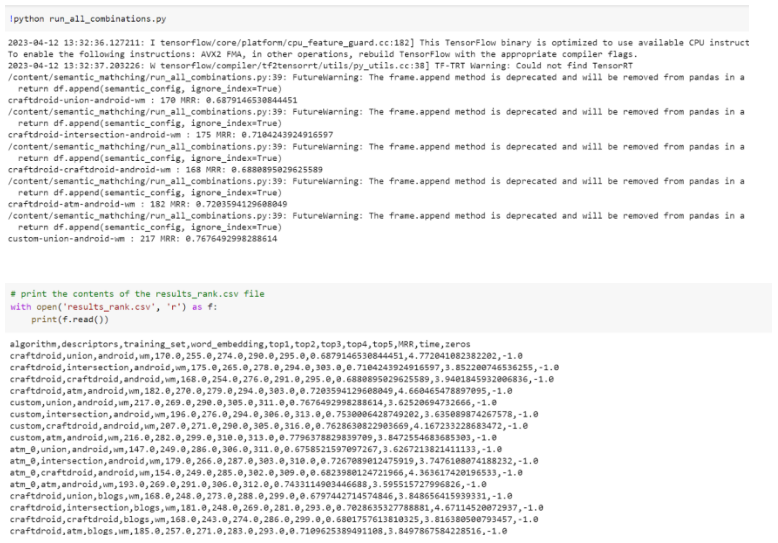


Figure 2 : Generalized architecture of the system

6 Results



```
!python run_all_combinations.py

2023-04-12 13:32:36.127211: I tensorflow/core/platform/cpu_feature_guard.cc:182] This TensorFlow binary is optimized to use available CPU instruct
To enable the following instructions: AVX2 FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
2023-04-12 13:32:37.203226: W tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Could not find TensorRT
/content/semantic_mathching/run_all_combinations.py:39: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a
  return df.append(semantic_config, ignore_index=True)
craftdroid-union-android-wm : 170 MRR: 0.6879146530844451
/content/semantic_mathching/run_all_combinations.py:39: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a
  return df.append(semantic_config, ignore_index=True)
craftdroid-intersection-android-wm : 175 MRR: 0.7104243924916597
/content/semantic_mathching/run_all_combinations.py:39: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a
  return df.append(semantic_config, ignore_index=True)
craftdroid-craftdroid-android-wm : 168 MRR: 0.6880895029625589
/content/semantic_mathching/run_all_combinations.py:39: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a
  return df.append(semantic_config, ignore_index=True)
craftdroid-atm-android-wm : 182 MRR: 0.7203594129608049
/content/semantic_mathching/run_all_combinations.py:39: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a
  return df.append(semantic_config, ignore_index=True)
custom-union-android-wm : 217 MRR: 0.7676492998288614
```

```
# print the contents of the results_rank.csv file
with open('results_rank.csv', 'r') as f:
    print(f.read())
```

```
algorithm,descriptors,training_set,word_embedding,top1,top2,top3,top4,top5,MRR,time,zeros
craftdroid,union,android,wm,170.0,255.0,274.0,290.0,295.0,0.6879146530844451,4.772041082382202,-1.0
craftdroid,intersection,android,wm,175.0,265.0,278.0,294.0,303.0,0.7104243924916597,3.852200746536255,-1.0
craftdroid,craftdroid,android,wm,168.0,254.0,276.0,291.0,295.0,0.6880895029625589,3.9401845932006836,-1.0
craftdroid,atm,android,wm,182.0,270.0,279.0,294.0,303.0,0.7203594129608049,4.660465478897095,-1.0
custom,union,android,wm,217.0,269.0,290.0,305.0,311.0,0.7676492998288614,3.62520694732666,-1.0
custom,intersection,android,wm,196.0,276.0,294.0,306.0,313.0,0.7530006428749202,3.635089874267578,-1.0
custom,craftdroid,android,wm,207.0,271.0,290.0,305.0,316.0,0.7628630822903669,4.167233228683472,-1.0
custom,atm,android,wm,216.0,282.0,299.0,310.0,313.0,0.7796378829839709,3.8472554683685303,-1.0
atm_0,union,android,wm,147.0,249.0,286.0,306.0,311.0,0.6758521597097267,3.6267213821411133,-1.0
atm_0,intersection,android,wm,179.0,266.0,287.0,303.0,310.0,0.7267089012475919,3.7476108074188232,-1.0
atm_0,craftdroid,android,wm,154.0,249.0,285.0,302.0,309.0,0.6823980124721966,4.363617420196533,-1.0
atm_0,atm,android,wm,193.0,269.0,291.0,306.0,312.0,0.7433114903446688,3.595515727996826,-1.0
craftdroid,union,blogs,wm,168.0,248.0,273.0,288.0,299.0,0.6797442714574846,3.848656415939331,-1.0
craftdroid,intersection,blogs,wm,181.0,248.0,269.0,281.0,293.0,0.7028635327788881,4.67114520072937,-1.0
craftdroid,craftdroid,blogs,wm,168.0,243.0,274.0,286.0,299.0,0.6801757613810325,3.816380500793457,-1.0
craftdroid,atm,blogs,wm,185.0,257.0,271.0,283.0,293.0,0.7109625389491108,3.8497867584228516,-1.0
```

Figure 3: Sample output of the system

Table : Best combinations as per MRR

	Algorithm	Descriptor	Training Set	MRR
1	custom	atm	Google play	0.795891209
2	custom	atm	standard	0.788772192
3	custom	atm	android	0.779637883

We now discuss the potential implications and limitations of this work.

7 Discussion

The table presented in the results section displays the optimal combinations based on Mean Reciprocal Rank (MRR), which will be listed in the result_rank.csv file. It is worth noting that the top three best combinations are highlighted in the table. According to the results, the combinations obtained from the Google Play, Standard, and Android training sets achieved the highest MRR scores of 0.79, 0.78, and 0.77, respectively. These findings suggest that the aforementioned training sets can effectively enhance the accuracy of the model, thus making them the most suitable choices for the given task.

8 CONCLUSION

In conclusion, UI testing is a crucial but expensive activity in software development. Test reuse approaches have emerged as a promising solution to reduce the cost of UI testing, but the success of these approaches relies heavily on the semantic matching of GUI events. The integration of semantic annotation of icons, widgets, and images can enhance the accuracy of semantic matching, and also improve the interpretability and maintainability of UI tests. This approach has the potential to reduce the overall cost and effort required for UI testing and improve the quality of software by enabling more comprehensive testing. These findings offer practical insights for software developers and researchers, highlighting the importance of image understanding of GUI widgets in test reuse.

Identifying and eliminating inconsequential and contradictory data automatically is an essential task that requires attention in the future. Also the potential of image understanding and semantic matching needs to be further explored to yield better results in the future for this field

REFERENCES

[1] Deka, B., Huang, Z., Franzen, C., Hibschman, J., Afergan, D., Li, Y., ... & Kumar, R. (2017, October). Rico: A mobile app dataset for building data-driven design applications. In Proceedings of the 30th annual ACM symposium on user interface software and technology (pp. 845-854).

[2] Mariani, L., Mohebbi, A., Pezzè, M., & Terragni, V. (2021, July). Semantic matching of gui events for test reuse: are we there yet?. In Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis (pp. 177-190).

[3] Leonardo Mariani, Ali Mohebbi, Mauro Pezze, & Valerio Terragni. (2021). Semantic Matching of GUI Events for Test Reuse: Are We There Yet? [Data set]. ACM/SIGSOFT International Symposium on Software Testing and Analysis (ISSTA). Zenodo. https://doi.org/10.5281/zenodo.4725222

[4] Talebipour, S., Zhao, Y., Dojcilović, L., Li, C., & Medvidović, N. (2021, November). UI test migration across mobile platforms. In *2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE)* (pp. 756-767). IEEE.

[5] Lin, J. W., Jabbarvand, R., & Malek, S. (2019, November). Test transfer across mobile apps through semantic mapping. In *2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)* (pp. 42-53). IEEE.

[6] Liu, S., Zhou, Y., Han, T., & Chen, T. (2022, December). Test Reuse based on Adaptive Semantic Matching across Android Mobile Applications. In *2022 IEEE 22nd International Conference on Software Quality, Reliability and Security (QRS)* (pp. 703-709). IEEE.

[7] Behrang, F., & Orso, A. (2019, November). Test migration between mobile apps with similar functionality. In *2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)* (pp. 54-65). IEEE.

[8] Sun, Xiaoyu, et al. "Mining android api usage to generate unit test cases for pinpointing compatibility issues." *37th IEEE/ACM International Conference on Automated Software Engineering*. 2022.

[9] Zhao, Y., Chen, J., Sejfia, A., Schmitt Laser, M., Zhang, J., Sarro, F., ... & Medvidovic, N. (2020, November). Fruiter: a framework for evaluating ui test reuse. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (pp. 1190-1201).

[10] Ardito, L., Bottino, A., Coppola, R., Lamberti, F., Manigrasso, F., Morra, L., & Torchiano, M. (2021). Feature matching-based approaches to improve the robustness of Android visual GUI testing. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, *31*(2), 1-32.

QuillBot

Scanned on: 10:4 April 14, 2023 UTC

10.7%

Overall similarity score

28

Results found

2468

Total words in text

	Word count
Identical	221
Minor Changes	43
Omitted	0

QuillBot

QuillBot

Scanned on: 10:4 April 14, 2023 UTC

Results

The results include any sources we have found in your submitted document that includes the following: identical text, minor changed text, paraphrased text.

Semantic matching of GUI events for test reuse: are we ther... *https://dl.acm.org/doi/abs/10.1145/3460319.3464827*	5%
Semantic matching of GUI events for test reuse: are we ther... *https://dl.acm.org/doi/10.1145/3460319.3464827*	5%
Semantic Matching of GUI Events for Test Reuse: Are We The... *https://valerio-terragni.github.io/assets/pdf/mariani-issta-2021.pdf* mariani-issta-2021.pdf	4%
2301.00530 *https://arxiv.org/pdf/2301.00530.pdf* 2301.00530.pdf	3%
Semantic Matching of GUI Events for Test Reuse: Are We The... *https://conf.researchr.org/details/issta-2021/issta-2021-technical-papers/...*	2%
Mauro Pezze' - Academia.edu *https://independent.academia.edu/Pezze*	2%
ISSTA 2021: Proceedings of the 30th ACM SIGSOFT Internatio... *https://www.sigsoft.org/resources/opentoc/ISSTA-21toc.html* ISSTA-21toc.html	2%
Advancing Automated Software Testing Through Test Reuse *https://escholarship.org/content/qt6wz7j1rs/qt6wz7j1rs_noSplash_f353cf5...* qt6wz7j1rs_noSplash_f353cf56b1280c7e2aa5650282f74a7c.pdf	2%

IDENTICAL

Text that is exactly the same.

MINOR CHANGES

Text that is nearly identical, yet a different form of the word is used. (i.e 'slow' becomes 'slowly')

Unsure about your report?

The results have been found after comparing your submitted text to online sources, open databases and the Copyleaks internal database. If you have any questions or concerns, please feel free to contact us atsupport@copyleaks.com

Click here to learn more about different types of plagiarism

QuillBot

Scanned on: 10:4 April 14, 2023 UTC

TransplantFix: Graph Differencing-based Code Transplantati... *https://dl.acm.org/doi/10.1145/3551349.3556893*	1%
Test migration between mobile apps with similar functional... *https://dl.acm.org/doi/abs/10.1109/ASE.2019.00016* ASE.2019.00016	1%
Bowen Xu *https://www.bowenxu.me/publications.html* publications.html	1%
6450ea28ebbc8437bc38775157818172-Paper-Conference.pdf *https://proceedings.neurips.cc/paper_files/paper/2022/file/6450ea28ebbc...* 6450ea28ebbc8437bc38775157818172-Paper-Conference.pdf	1%
Mathematics | Free Full-Text | Semantic Similarity-Based M... *https://www.mdpi.com/2227-7390/11/3/527*	1%
Modern Code Reviews - A Survey of Literature and Practice |... *https://dl.acm.org/doi/10.1145/3585004*	1%
ISSTA 2021 - Technical Papers - ISSTA 2021 *https://2021.icse-conferences.org/track/issta-2021/issta-2021-technical-pa...*	1%
ASE4Games 2021 Workshop Summary | ACM SIGSOFT Softw... *https://dl.acm.org/doi/abs/10.1145/3520273.3520277*	1%

QuillBot

QuillBot

Scanned on: 10:4 April 14, 2023 UTC

Publications – Data Link Institute of Business & Technology *https://datalink.edu.gh/publications/*	1%
GaoFangMemonISSRE2015.pdf *https://www.cs.umd.edu/users/atif/papers/GaoFangMemonISSRE2015.pdf* GaoFangMemonISSRE2015.pdf	1%
Automated Software Engineering (ASE), 2019 34th IEEE/ACM ... *https://journals.scholarsportal.info/browse/26431572/v2019inone*	1%
Feature Matching-based Approaches to Improve the Robust... *https://figshare.com/articles/online_resource/Feature_Matching-based_Ap...*	1%
FrUITeR: A Framework for Evaluating UI Test Reuse *http://www0.cs.ucl.ac.uk/staff/F.Sarro/resource/papers/FSE2020-FrUITeR.pdf* FSE2020-FrUITeR.pdf	1%
ASE4Games 2021 Workshop Summary | ACM SIGSOFT Softw... *https://dl.acm.org/doi/10.1145/3520273.3520277*	1%
Evaluating State-of-the-Art Visual Question Answering Mod... *https://www.jsr.org/hs/index.php/path/article/view/2446*	1%
PRIORITY: An Intelligent Problem Indicator Repository | Pro... *https://dl.acm.org/doi/10.1145/3578527.3578533*	1%

QuillBot

QuillBot

Scanned on: 10:4 April 14, 2023 UTC

Publications | Alejandro Cuevas *https://alejandrocuevas.me/publications/*	1%
37th IEEE/ACM International Conference on Automated Soft... *https://www.dote.osd.mil/Home/mid/33173/itemid/1834/ctl/Details/Show...*	1%
ASE 2022 : 37th IEEE/ACM International Conference on Auto... *http://www.wikicfp.com/cfp/servlet/event.showcfp?eventid=149330*	1%
ASE 2022 *https://conf.researchr.org/home/ase-2022*	1%

Scanned Text

Your text is highlighted according to the plagiarism types that where found, as shown above.

● IDENTICAL ● MINOR CHANGES

Image Understanding of GUI Widgets for Test Reuse
Swarnalatha P pswarnalatha@vit.ac.in Vellore Institute of Technology, India
Yash Mishra yash.mishra2020@vitstudent.ac.in Vellore Institute of Technology, India
Prazwal Lamsal prazwal.lamsal2020@vitstudent.ac.in Vellore Institute of Technology, India
Prithak Gajurel prithak.gajurel2020@vitstudent.ac.in Vellore Institute of Technology, India
ABSTRACTUI testing is a crucial but costly activity in software development. Test reuse approaches have recently emerged as a promising solution to reduce the cost of UI testing. These approaches automatically transfer human-designed GUI tests from a source application to a target application with similar functionalities, exploiting semantic similarity among textual information of GUI widgets. However, the success of test reuse approaches hinges on the semantic matching of GUI events. This paper presents the integration of semantic annotation of icons, widgets and images for semantic matching of GUI events. Our findings demonstrate the importance of image understanding of GUI widgets in test reuse and offer practical insights for software developers and researchers. In addition to improving semantic matching, the integration of semantic annotation of icons, widgets, and images can also enhance the interpretability and maintainability of UI tests. By adding semantic information to the test cases, it becomes easier to understand and modify the tests, reducing the overall cost and effort required for UI testing in software development. Furthermore, this approach can potentially improve the overall quality of the software by enabling more comprehensive testing, leading to better user experience and fewer bugs.
INTRODUCTIONUI testing is a crucial but costly activity in software development. Test reuse approaches have recently emerged as a promising solution to reduce the cost of UI testing. These approaches automatically transfer human-designed GUI tests from a source application to a target application with similar functionalities, exploiting semantic similarity among textual information of GUI widgets. However, the success of test reuse approaches hinges on the semantic matching of GUI events. This paper presents the first empirical study on semantic matching of GUI events, which involves 253 configurations of the semantic matching, 337 unique queries, and 8,099 distinct GUI events. Our study yields several key findings that shed light on how to improve the semantic matching of test reuse approaches. Additionally, we propose SemFinder, a novel semantic matching algorithm that outperforms existing solutions. Finally, we identify several interesting research directions for future work in this area. Our findings demonstrate the importance of image understanding of GUI widgets in test reuse and offer practical insights for software developers and researchers.GUI test reuse is a promising approach for generating meaningful test cases for GUI applications. By leveraging the observation that many GUI applications share similar functionalities, automatic approaches can migrate GUI tests across apps by mapping semantically similar GUI events. This approach addresses the limitations of current GUI test generators, which often generate semantically meaningless tests and rely on implicit oracles.
Now, in the context of image understanding of GUI widgets for test reuse, it is possible to leverage techniques such as computer vision and deep learning to automatically recognize and match GUI widgets across different applications. By analyzing the visual features and properties of GUI widgets, we can identify semantically similar widgets across different applications and use them to generate meaningful test cases for the target application.
Furthermore, by incorporating image understanding into the GUI test reuse approach, we can potentially address the limitations of current test generators and improve the accuracy and effectiveness of generated test cases. For example, by using image understanding to identify and match semantically

similar GUI widgets, we can ensure that the generated test cases properly exercise the relevant functionalities of the target application and reveal faults that might be missed by existing test generators. Overall, image understanding of GUI widgets for test reuse is a promising research direction that has the potential to significantly improve the quality and efficiency of GUI test generation for software applications.
ScopeThe scope of this topic is broad and encompasses various aspects of computer science and software engineering. One key area of focus is computer vision, which involves developing algorithms and techniques for analyzing and understanding visual data, such as images and videos. In the context of GUI test reuse, computer vision can be used to automatically recognize and match GUI widgets across different applications.
Another important area of focus is deep learning, which is a subfield of machine learning that involves training artificial neural networks to perform complex tasks, such as image recognition and natural language processing. In the context of GUI test reuse, deep learning techniques can be used to train models that can automatically identify and match semantically similar GUI widgets.
Finally, the scope of GUI testing is also relevant to this topic, as it involves developing techniques and tools for testing graphical user interfaces. In the context of GUI test reuse, this includes the development of methods and algorithms for automatically generating test cases for GUI applications based on existing test cases from other applications. Overall, the scope of image understanding of GUI widgets for test reuse is an interdisciplinary field that involves computer vision, deep learning, and GUI testing to improve the efficiency and effectiveness of software testing.
.

3 ObjectivesTo develop algorithms and techniques for automatically recognizing and matching GUI widgets across different applications, which can help in reusing existing test cases for new applications.

To investigate the use of deep learning techniques for automatically identifying semantically similar GUI widgets across different applications, which can help in generating more meaningful and effective test cases.

To improve the efficiency and effectiveness of GUI testing by developing methods for automatically generating test cases based on existing test cases from other applications.

To address the limitations of current GUI test generators, which often generate semantically meaningless GUI tests that miss many relevant behaviors of the application under test.

To explore the reuse of GUI tests across similar applications as an alternative way to automatically generate GUI tests, which can help in generating semantically meaningful GUI tests that properly exercise the functionalities of the target application.

To develop methods and algorithms for adapting semantically relevant oracle assertions to the target application when reusing GUI tests, which can help in addressing the main limitations of GUI test generators.
4 Software SpecificationsAs The major components of the system are the dataset used and the framework used for judging the different approaches for suggesting GUI test transfer from one trace of events to another. The dataset used is 'Rico: A Mobile App Dataset for Building Data-Driven Design Applications'[1]. It possesses over 70000 unique UI screens from over 9300 apps. It has defined traces for different UI events. It has also provided semantic annotations to all the GUI elements present in the UI screens along with json files describing the hierarchical relation between the objects. All of these features make it ideal for use.

Figure 1: Example of semantic annotation of UI Screen in Rico dataset.
The other component of the system is the framework used in [2]. The framework analyzes the usage of ATM and CraftDroid for test transfer and also proposes a custom algorithm called SemFinder which tries to perform the same task by taking the help of semantic matching of GUI events. The framework was obtained online at [3]. The framework and dataset are implemented using python on Google Colab
5 Methodology and System Architecture
The previous work on semantic matching for GUI event test transfer discusses semantic annotation based only on text. However, including the semantic annotations of GUI widgets and icons could probably boost the performance of certain techniques. The main objective of this work is to integrate semantic annotations of GUI widgets into the existing system for judging the variation in performance caused by it.

Dataset

The Rico dataset[1] has a wide array of data samples of UI screenshots and related information such as trace of events, UI layout vectors, semantic annotations and descriptions of hierarchical relationships. The presence of these features in the dataset is essential for the task as information regarding different UI elements need to be collected step-wise as per the trace of the event. Also the semantic annotation of icons and widgets allows for easy integration of semantics of images.

Extraction of data

The relevant data was extracted from the dataset. Essential data (event_index, label, type) as well as optional data (text, id, content_desc, hint, parent_text, sibling_text, activity,atm_neighbor, file_name) were all taken from the dataset via mining of the dataset. The extraction of relevant data from the dataset is a critical step in data mining and is essential for obtaining meaningful insights and patterns from the data. By selecting essential data such as event_index, label, and type, the extracted data can be used for a variety of purposes, such as training machine learning models, detecting anomalies, and understanding user behavior.

Semantic Matching Framework

The model provided at [3] has readily been trained on various state of the art models like GLOVE, word2vec and so on. The same framework was utilized for testing the earlier obtained data as well after it was adjusted into suitable input format. The framework judged the performance of semantic matching of different events by the different test migration algorithm, descriptor algorithm, training set that was used to train the model and word embedding model.
The use of state-of-the-art models like GLOVE and word2vec in the training of the model provides a solid foundation for semantic matching of different events. By leveraging pre-trained word embeddings, the model can capture the semantic similarity between textual information of GUI widgets, enabling effective transfer of human-designed GUI tests from a source application to a target application with similar functionalities.
The testing of the earlier obtained data after adjustment into a suitable input format provides valuable insights into the performance of the framework. By evaluating the performance of semantic matching using different test migration algorithms, descriptor algorithms, training sets, and word embedding models, it becomes possible to identify the best approach for specific use cases.
The framework presented in [3] provides a robust and flexible solution for UI test reuse, leveraging state-of-the-art models and algorithms to achieve high accuracy in semantic matching. The insights gained from the evaluation of the framework can be applied to various domains, improving the efficiency and effectiveness of UI testing in software development.

Evaluation Metrics

Mean Reciprocal Rank (MRR) has been used for evaluation as a metric which has been used in [2] as well. The advantage of using MRR as an evaluation metric is that it takes into account the order of the results, rather than just whether or not the relevant item was present in the list. In other words, MRR rewards algorithms that place relevant items at the top of the list, and penalizes algorithms that bury relevant items lower down in the list.

Figure 2 : Generalized architecture of the system

6 Results

Figure 3: Sample output of the system

Table : Best combinations as per MRR

	Algorithm	Descriptor	Training Set	MRR
1	custom	atm	Google play	0.795891209
2	custom	atm	standard	0.788772192

3
custom
atm
android
0.779637883
We now discuss the potential implications and limitations of this work.
7 DiscussionThe table presented in the results section displays the optimal combinations based on Mean Reciprocal Rank (MRR), which will be listed in the result_rank.csv file. It is worth noting that the top three best combinations are highlighted in the table. According to the results, the combinations obtained from the Google Play, Standard, and Android training sets achieved the highest MRR scores of 0.79, 0.78, and 0.77, respectively. These findings suggest that the aforementioned training sets can effectively enhance the accuracy of the model, thus making them the most suitable choices for the given task.
CONCLUSIONIn conclusion, UI testing is a crucial but expensive activity in software development. Test reuse approaches have emerged as a promising solution to reduce the cost of UI testing, but the success of these approaches relies heavily on the semantic matching of GUI events. The integration of semantic annotation of icons, widgets, and images can enhance the accuracy of semantic matching, and also improve the interpretability and maintainability of UI tests. This approach has the potential to reduce the overall cost and effort required for UI testing and improve the quality of software by enabling more comprehensive testing. These findings offer practical insights for software developers and researchers, highlighting the importance of image understanding of GUI widgets in test reuse.
Identifying and eliminating inconsequential and contradictory data automatically is an essential task that requires attention in the future. Also the potential of image understanding and semantic matching needs to be further explored to yield better results in the future for this field
REFERENCESDeka, B., Huang, Z., Franzen, C., Hibschman, J., Afergan, D., Li, Y., ... & Kumar, R. (2017, October). Rico: A mobile app dataset for building data-driven design applications. In Proceedings of the 30th annual ACM symposium on user interface software and technology (pp. 845-854).
Mariani, L., Mohebbi, A., Pezzè, M., & Terragni, V. (2021, July). Semantic matching of gui events for test reuse: are we there yet?. In Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis (pp. 177-190).
Leonardo Mariani, Ali Mohebbi, Mauro Pezze, & Valerio Terragni. (2021). Semantic Matching of GUI Events for Test Reuse: Are We There Yet? [Data set]. ACM/SIGSOFT International Symposium on Software Testing and Analysis (ISSTA). Zenodo. https://doi.org/10.5281/zenodo.4725222
Talebipour, S., Zhao, Y., Dojcilović, L., Li, C., & Medvidović, N. (2021, November). UI test migration across mobile platforms. In 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE) (pp. 756-767). IEEE.
Lin, J. W., Jabbarvand, R., & Malek, S. (2019, November). Test transfer across mobile apps through semantic mapping. In 2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE) (pp. 42-53). IEEE.
Liu, S., Zhou, Y., Han, T., & Chen, T. (2022, December). Test Reuse based on Adaptive Semantic Matching across Android Mobile Applications. In 2022 IEEE 22nd International Conference on Software Quality, Reliability and Security (QRS) (pp. 703-709). IEEE.
Behrang, F., & Orso, A. (2019, November). Test migration between mobile apps with similar functionality. In 2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE) (pp. 54-65). IEEE.
Sun, Xiaoyu, et al. "Mining android api usage to generate unit test cases for pinpointing compatibility issues." 37th IEEE/ACM International Conference on Automated Software Engineering. 2022.
Zhao, Y., Chen, J., Sejfia, A., Schmitt Laser, M., Zhang, J., Sarro, F., ... & Medvidovic, N. (2020, November). Fruiter: a framework for evaluating ui test reuse. In Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (pp. 1190-1201).
Ardito, L., Bottino, A., Coppola, R., Lamberti, F., Manigrasso, F., Morra, L., & Torchiano, M. (2021). Feature matching-based approaches to improve the robustness of Android visual GUI testing. ACM Transactions on Software Engineering and Methodology (TOSEM), 31(2), 1-32.

Project Title Name:

Image Understanding of GUI Widgets and Test Resuse

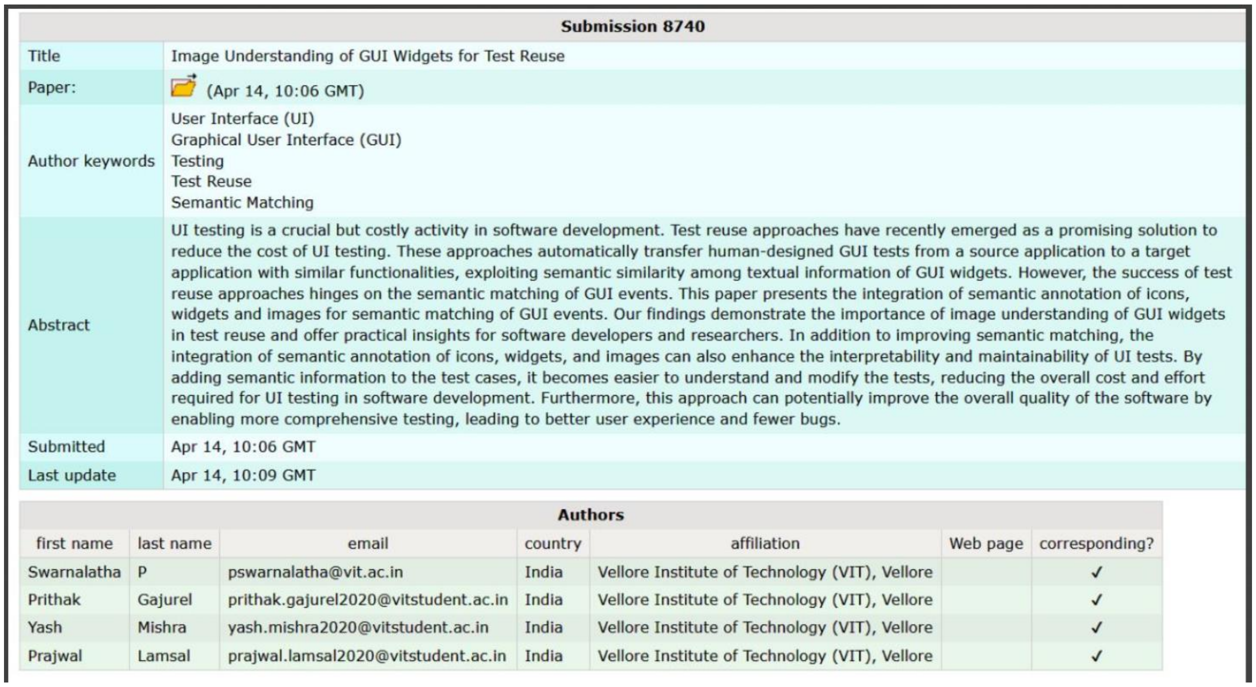
Team Members:

Prajwal Lamsal(20BCE2901)

Yash Mishra(20BCE2773)

Prithak Gajurel(20BCE2921)

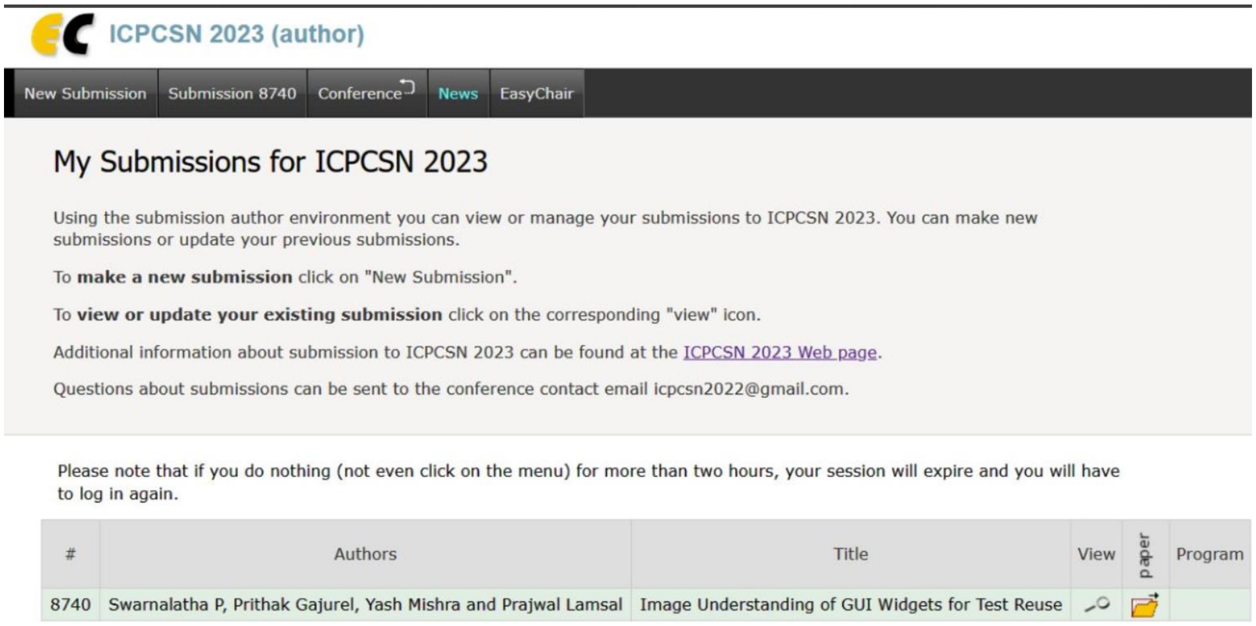
Submisson proof:



Submission 8740	
Title	Image Understanding of GUI Widgets for Test Reuse
Paper:	(Apr 14, 10:06 GMT)
Author keywords	User Interface (UI) Graphical User Interface (GUI) Testing Test Reuse Semantic Matching
Abstract	UI testing is a crucial but costly activity in software development. Test reuse approaches have recently emerged as a promising solution to reduce the cost of UI testing. These approaches automatically transfer human-designed GUI tests from a source application to a target application with similar functionalities, exploiting semantic similarity among textual information of GUI widgets. However, the success of test reuse approaches hinges on the semantic matching of GUI events. This paper presents the integration of semantic annotation of icons, widgets and images for semantic matching of GUI events. Our findings demonstrate the importance of image understanding of GUI widgets in test reuse and offer practical insights for software developers and researchers. In addition to improving semantic matching, the integration of semantic annotation of icons, widgets, and images can also enhance the interpretability and maintainability of UI tests. By adding semantic information to the test cases, it becomes easier to understand and modify the tests, reducing the overall cost and effort required for UI testing in software development. Furthermore, this approach can potentially improve the overall quality of the software by enabling more comprehensive testing, leading to better user experience and fewer bugs.
Submitted	Apr 14, 10:06 GMT
Last update	Apr 14, 10:09 GMT

Authors

first name	last name	email	country	affiliation	Web page	corresponding?
Swarnalatha	P	pswarnalatha@vit.ac.in	India	Vellore Institute of Technology (VIT), Vellore		✓
Prithak	Gajurel	prithak.gajurel2020@vitstudent.ac.in	India	Vellore Institute of Technology (VIT), Vellore		✓
Yash	Mishra	yash.mishra2020@vitstudent.ac.in	India	Vellore Institute of Technology (VIT), Vellore		✓
Prajwal	Lamsal	prajwal.lamsal2020@vitstudent.ac.in	India	Vellore Institute of Technology (VIT), Vellore		✓



ICPCSN 2023 (author)

My Submissions for ICPCSN 2023

Using the submission author environment you can view or manage your submissions to ICPCSN 2023. You can make new submissions or update your previous submissions.

To **make a new submission** click on "New Submission".

To **view or update your existing submission** click on the corresponding "view" icon.

Additional information about submission to ICPCSN 2023 can be found at the ICPCSN 2023 Web page.

Questions about submissions can be sent to the conference contact email icpcsn2022@gmail.com.

Please note that if you do nothing (not even click on the menu) for more than two hours, your session will expire and you will have to log in again.

#	Authors	Title	View	paper	Program
8740	Swarnalatha P, Prithak Gajurel, Yash Mishra and Prajwal Lamsal	Image Understanding of GUI Widgets for Test Reuse	🔍	📂	

ICPCSN 2023 submission 8740

ICPCSN 2023 <icpcsn2023@easychair.org>
3:38 PM

To: Prajwal Lamsal

Dear authors,

We received your submission to ICPCSN 2023 (3rd International
Conference on Pervasive Computing and Social Networking):

Authors : Swarnalatha P, Prithak Gajurel, Yash Mishra and Prajwal Lamsal
Title : Image Understanding of GUI Widgets for Test Reuse
Number : 8740

The submission was uploaded by Prithak Gajurel
<prithak.gajurel2020@vitstudent.ac.in>. You can access it via the
ICPCSN 2023 EasyChair Web page

https://www.google.com/url?q=https://easychair.org/conferences/?conf%3Dicpcsn2023&source=gmail-imap&ust=1682071690000000&usg=AOvVaw0aReYo89AY-_1zusBcFNfl

Thank you for submitting to ICPCSN 2023.

Best regards,
EasyChair for ICPCSN 2023.