**Abstract**

Importance of UI Testing: The paper highlights the significance of testing UI after its development. It emphasizes that UI testing is an essential activity to ensure the quality, usability, and functionality of the user interface.

Cost and Complexity of Test Development: The paper acknowledges the complexity of developing new tests for UI, which can be a costly process. It recognizes that creating tests specific to each UI scenario is an expensive task and presents the need for approaches that allow for test reuse.

Test Reusability Approaches: The paper mentions different approaches that have been developed to enable test reuse in similar UI scenarios. These approaches utilize various information available in the GUI to determine the transferability of tests from one UI to another.

Integration of Semantic Knowledge: The paper highlights the importance of semantic knowledge in understanding the similarities between GUI events. It suggests that utilizing semantic information can aid in matching GUI events and improving the transferability of tests.

Leveraging Textual Information: The paper discusses the promising results achieved by using textual information available in the GUI to enhance test reuse. It suggests that incorporating semantic meanings derived from textual information can be beneficial in the task of matching GUI events.

Unexplored Potential of Image Understanding: The paper introduces the idea of leveraging semantic information obtained from images and icons in GUI for test reuse. It argues that image understanding of GUI widgets can play a significant role in improving test matching and offers practical insights for software developers and researchers.

Enhanced Interpretability and Maintainability: The paper highlights that integrating semantic annotation of icons, widgets, and images in UI tests can not only improve semantic matching but also enhance the interpretability and maintainability of the tests. This implies that developers and testers can better understand and maintain the tests over time.

**Introduction**

Cost and Effort of Manual Test Creation: The paper acknowledges the high costs and manual effort involved in creating test scenarios for mobile applications. Manual test case creation becomes particularly challenging due to the rapid app development lifecycle, making it time-consuming and tedious.

Automatic Test Case Generation: Researchers are working on techniques to automate test case generation in order to reduce time consumption and manual efforts. Automated test case generation can help streamline the testing process and improve efficiency.

Challenges of Automated Test Case Generation: Despite advancements in automated test case generation, the industry still prefers manually created test cases. The paper outlines three main reasons for this preference: (1) Lack of context-aware text inputs, (2) Inability to generate expressive tests, and (3) Absence of test oracles.

Semantic Similarities and Test Migration: The paper highlights the potential of leveraging semantic similarities between different graphical user interfaces (GUIs) to enable effective test migration from one GUI to another. By identifying semantically related events using word embedding techniques on textual descriptors of GUI widgets, test migration becomes feasible.

Impact of Semantic GUI Event Matching: The success of semantic GUI event matching significantly affects the overall effectiveness of test reuse. Matching events between the source and target tests is driven by semantic matching, which plays a crucial role in the transferability of tests.

Incorporating Images and Icons as Semantic Descriptors: The paper suggests that besides textual descriptors, the images and icons present in the UI can be employed as semantic descriptors. These visual elements carry additional information that can aid in test reuse. Machine learning methods have been developed to extract information from widgets and icons.

Evaluation of Semantic Matching with Images and Icons: The paper proposes utilizing the same framework employed in a previous study to evaluate the performance differences in various dimensions when using semantically annotated icons and images. The study utilizes the Rico dataset, which provides semantic annotations for UI components, text buttons, and icon classes. This evaluation aims to explore the impact of integrating images and icons in test reuse methodologies.

**Related Work**

Rise in Test Reusability Research: The research on the reusability of tests for mobile application testing has been increasing due to the complexity of UI testing. Techniques like AppFlow, ATM, CraftDroid, AdaptDroid, and TRASM have been proposed to enable test migration and adaptation for different apps on the Android platform.

Test Migration within and across Platforms: Techniques like ATM, CraftDroid, and TRASM focus on test migration within apps on the same platform (Android). MAPIT, on the other hand, enables test migration between sibling apps across different platforms by building models of the source and target UI and performing mapping between them.

Testing the Test Reusability Techniques: Evaluating and testing the effectiveness of test reusability techniques is crucial. FrUITeR is a framework that evaluates UI test reuse techniques, while a framework introduced in [6] enables understanding the reasons behind better or worse performance of these techniques by analyzing different attributes and word embedding models.

Classification of Widgets and Icons: Classification of widgets and icons plays a significant role in understanding their semantics. Techniques like IconIntent, LabelDroid, and ReDraw utilize computer vision and machine learning to classify widgets and icons, enabling various applications such as accessibility enhancement and intent generation.

Importance of Semantic Matching: SemFinder, a new semantic matching algorithm, is introduced in [6]. It focuses on the importance of different attributes of events in test migration. However, the integration of semantic understanding of icons, images, and widgets as additional information remains unexplored.

Framework for Studying the Impact of Semantic Descriptors: The paper proposes utilizing a framework similar to [6] to study the impact of incorporating semantic annotations of icons, images, and widgets as descriptors. This framework allows for evaluating the change in performance when using additional visual information along with textual information in test migration and adaptation.

Potential Benefits of Semantic Descriptors: By leveraging semantic understanding of icons, images, and widgets, it is anticipated that the performance of semantic matching and test reuse techniques can be improved. Exploring the integration of visual information can provide valuable insights into the impact of image and icon semantics on the overall effectiveness of test migration and adaptation.

**Methodology**

Dataset of GUI: The study utilizes the Rico dataset, which contains GUI event traces and corresponding UI screenshots from various Android applications. The dataset provides hierarchies and semantic annotations for UI components, including icons, text buttons, and general components.

Corpus of Documents: The study considers three corpora of English documents: Blog Authorship Corpus, User Manuals of Android apps, and Apps Descriptions from the Google Play Store. These corpora are used for building word embedding models and providing descriptions for word and sentence semantics.

Word Embeddings: Various word embedding techniques are employed, including Word Mover's distance, Word2vec, Neural Network Language Model, Universal Sentence Encoder, GlobalVectors (Glove), and FastText. These techniques convert words and sentences into continuous vector representations, capturing their semantic relationships.

Event Descriptor Extractor: The event descriptor extractor processes GUI event traces and extracts relevant attributes from hierarchy files and semantic annotations. These attributes include text, ID, content description, hint, activity, parent text, sibling text, fillable neighbor, and more. The framework categorizes these attributes into different groups based on the algorithms used.

Semantic Matching Algorithm: The study utilizes three semantic matching algorithms: ATM, CraftDroid, and SemFinder. These algorithms determine how to generate target test cases by calculating similarity scores between source and target event descriptors. They leverage word embedding models to compute semantic similarity scores and utilize aggregation functions (average or sum) to determine overall similarity.

Comparison of Semantic Matching Algorithms: The differences among the algorithms lie in the attributes they compare, the aggregation of similarity scores, and the aggregation of word-level word embedding models. ATM and CraftDroid have restrictions on attribute types, while SemFinder considers all attributes. Each algorithm applies different aggregation functions to calculate similarity scores, such as maximum, sum, or average.

**Experiment**

Experimental Setup: The study conducted experiments using different combinations of corpora (Manuals, Blogs, Google-play) and word embedding techniques (Word2vec, WMD, Glove, Fast) to create 12 word embedding models. Preprocessing steps were applied before applying the word embedding techniques, and pretrained models provided by the authors were used.

Syntactic Similarity Approaches: Two syntactic approaches, edit-distance based similarity (ES) and Jaccard Similarity index (JS), were considered for computing the syntactic similarity of words/sentences. These approaches do not utilize the corpora and were not combined with them.

Evaluation Metrics: The evaluation of semantic matching effectiveness used two metrics: Mean Reciprocal Rank (MRR) and the proportion of queries with the correct answer ranked at one (TOP1). MRR calculates the average of the reciprocal ranks of the queries, considering the position of the first correct answer. TOP1 measures the percentage of queries where the correct answer is at the top position.

MRR Calculation: MRR is based on reciprocal ranks, where the rank of the first correct answer is considered. The reciprocal ranks range from 1 for the first place to 1/n for the nth place, where n is the total number of candidate events. The MRR is the average of these reciprocal ranks.

TOP1 Calculation: TOP1 calculates the ratio of queries where the ground truth event is ranked first in the returned list of events. It indicates the percentage of queries where the correct answer is at the top position.

Evaluation of Semantic Matching: The metrics MRR and TOP1 are used to assess the effectiveness of semantic matching in retrieving the correct answer among the candidate events. These metrics provide insights into the performance of different combinations of corpora and word embedding techniques in the framework.

**Result and Discussion**

Performance Improvement: The integration of semantic annotations of images, icons, and widgets resulted in improvements in Mean Reciprocal Rank (MRR) and Top1 values, although the increments in performance were relatively small.

Higher MRR with Images: The inclusion of image semantics led to higher MRR values across all configurations. The range of MRR values with images (0.491 to 0.745) was higher than that without images (0.471 to 0.696), indicating that leveraging semantic information from images contributes to a better understanding and matching of GUI events.

Higher Top1 with Images: Similarly, the inclusion of image semantics resulted in higher Top1 values compared to configurations without image semantics. The range of Top1 values with images (0.071 to 0.455) surpassed the range without images (0.045 to 0.409), further supporting the idea that considering image semantics improves the accuracy of matching GUI events.

Most Improved Configuration (MRR): The configuration [ATM\_a, Intersection, Google Play, WM] demonstrated the most significant improvement in MRR value after integrating semantic annotation of images, icons, and widgets, with an increment of 0.062. This configuration showcases the effectiveness of specific combinations of event descriptors and semantic matching algorithms when incorporating image semantics.

Most Improved Configuration (Top1): The configuration [SemFinder, Union, Standard, USE] showed the largest improvement in Top1 value, with an increase of 0.107. This highlights the effectiveness of this specific combination in achieving accurate matching results when considering image semantics.

Component Distribution Analysis: Analyzing the distribution of component instances for different percentiles provides insights into the utilization of various components in the top-performing models. For example, in the first percentile, all entries employed the ATM algorithm, indicating its consistent effectiveness in achieving higher MRR values. Examining the distribution of other components can offer further understanding of their impact on the semantic matching performance.

**Conclusion**

Impact of Semantic Annotations: The study investigated the impact of incorporating semantic annotations of images, icons, and widgets on the performance of semantic matching of GUI events. The results showed that leveraging image semantics resulted in improvements in the Mean Reciprocal Rank (MRR) and Top1 values, indicating enhanced accuracy in matching GUI events.

Benefits of Image Semantics: The comparison between configurations with and without image semantics revealed that all configurations benefited from the inclusion of image semantics. The configurations incorporating image semantics achieved higher MRR and Top1 values, indicating improved matching performance. This finding highlights the importance of considering visual information, such as images and icons, in semantic matching of GUI events.

Incremental Improvements: While the incorporation of image semantics led to improvements in matching performance, the observed increments were relatively small. This suggests that image semantics alone may not be sufficient for achieving highly accurate and robust semantic matching. Other factors, such as text-based semantics and syntactic approaches, should also be taken into account to further enhance matching accuracy.

Consideration of Multiple Factors: The study suggests that a comprehensive approach to semantic matching should consider multiple factors, including image semantics, text-based semantics, and syntactic approaches. By integrating these various dimensions, a more accurate and robust semantic matching of GUI events can be achieved.

Future Research Directions: The paper proposes future research directions, such as real-time identification and classification of icons, images, and widgets from screenshots using machine learning techniques. This approach can provide additional valuable information for semantic matching and can be integrated into the process of migrating test cases along with other relevant information.

Practical Implications: The findings of this study have practical implications for researchers and practitioners in the field of UI testing. By incorporating image semantics, developers and testers can enhance the accuracy of semantic matching in GUI event recognition, leading to improved test case migration and overall test reuse.