УДК 004.89 **Информационные технологии**

Сарычева Юлия Юрьевна, *студент-магистр, Калужский филиал ФГБОУ ВО «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»*

Белов Юрий Сергеевич, *к.ф.-м.н., доцент, Калужский филиал ФГБОУ ВО «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»*

ИЗМЕРЕНИЕ ТЕСТОВОГО ПОКРЫТИЯ ГЕНЕРАТОРА ВХОДНЫХ ДАННЫХ ДЛЯ GUI МОБИЛЬНЫХ ПРИЛОЖЕНИЙ НА БАЗЕ ОС ANDROID

Аннотация: Модель необходимо оценивать для того, чтобы определить, будет ли она хорошо прогнозировать цель на новых и будущих данных.

Чтобы количественно определить и измерить качество генерации используется тестовое покрытие. Тестовое покрытие – одна из метрик оценки качества тестирования, которая оценивает плотность покрытия тестами кода приложения либо требования. Покрытие кода — это измерение, которое отслеживает, какая часть исходного кода программы или байтового кода выполняется во время тестового прогона.

Ключевые слова: тестовое покрытие, тестирование, мобильное приложение, автоматизированное тестирование.

Abstract: The model needs to be evaluated to determine if it will perform well on the target given new and future data.

Test coverage is used to quantify and measure generation quality. Test coverage is one of the metrics for assessing the quality of testing, which evaluates the density of test coverage of application code or requirements. Code coverage is a measurement that tracks how much of a program's source code or byte code is executed during a test

run.

Keywords: test coverage, testing, mobile application, automated testing.

Введение. Модель необходимо оценивать для того, чтобы определить, будет ли она хорошо прогнозировать цель на новых и будущих данных. Поскольку будущие экземпляры имеют неизвестные целевые значения, необходимо проверять показатель точности модели машинного обучения на данных, для которых уже известен целевой ответ, и использовать эту оценку в качестве прокси для прогнозируемой точности будущих данных [1].

Для экспериментов по тестированию реальных приложений Android был использован компьютер с эмулятором Android. Для оценки тестирования было использовано 68 приложений Android с открытым исходным кодом, полученных из AndroTest, широко используемого набора данных для оценки генераторов тестовых входных данных Android [2].

Представим модель, автоматизированный генератор входных данных с графическим интерфейсом, которая может изучать, как люди взаимодействуют с мобильными приложениями, а затем использовать ее для управления генерацией входных данных, имитируя поведение человека. Обладая знаниями и моделью, извлеченными из истории человеческого взаимодействия, модель может расставить приоритеты возможных взаимодействий с графическим интерфейсом в соответствии с их важностью с точки зрения пользователя, тем самым генерируя входные данные, которые могут достичь большего охвата.

Структура модели. Ядром рассматриваемой модели является модель глубокой нейронной сети (DNN) [3], которая предсказывает, какие входные данные с большей вероятностью будут генерироваться пользователями. Входные данные модели — это текущее состояние пользовательского интерфейса, а также самые последние переходы пользовательского интерфейса, представленные в виде пачки изображений, а выходные данные — прогнозируемое распределение возможных следующих действий, включая тип действия и соответствующие координаты местоположения на экране [4].

Сравнивая предсказанное распределение со всеми возможными действиями на странице пользовательского интерфейса, модель может назначить вероятность каждого действия.

Эксперимент. Был проведен эксперимент, в ходе которого модель была обучена и протестирована на существующих трассировках человеческого взаимодействия, чтобы оценить, способна ли модель узнать, как люди взаимодействуют с приложениями.

Все полученные результаты были разбиты по группам, в зависимости от интерактивных элементов, имеющихся в приложении:

1) приложения, где есть взаимодействия с клавиатурой;

2) приложения, где есть взаимодействия с картой;

3) приложения, где есть выпадающий список или же чек-боксы;

4) приложения, где нет интерактивных элементов, кроме кнопок.

Если же в приложении имеется несколько элементов, то относим приложение к более приоритетной группе. Например, если в приложении есть и взаимодействие и с картой и клавиатурой, то относим к группе приложений, где есть взаимодействие с клавиатурой. Результаты представлены в таблицах 1-4 и на рисунке 1.

Таблица 1. Максимальное покрытие для каждого приложений, где есть взаимодействие с клавиатурой, полученная с помощью Monkey и моделью

Приложение, №	Monkey	Модель
62	17%	33%
32	15%	19%
28	14%	22%
51	6%	95%
46	19%	31%
33	38%	59%
20	18%	18%
63	25%	35%

61	16%	19%
59	4%	5%
60	21%	29%
57	19%	20%
54	16%	17%
42	4%	6%
Средний коэффициент покрытия инструментом	16,57%	29,14%

Таблица 2. Максимальное покрытие для каждого приложений, где есть взаимодействия с картой, полученная с помощью Monkey и моделью

Приложение, №	Monkey	Модель
4	24%	27%
7	19%	27%
11	20%	21%
15	13%	13%
23	14%	13%
24	15%	9%
64	33%	44%
17	15%	17%
50	23%	30%
48	20%	24%
43	19%	22%
35	24%	16%
38	31%	29%
39	29%	27%
Средний коэффициент покрытия инструментом	21,36%	24,71%

Таблица 3. Максимальное покрытие для каждого приложений, где есть выпадающий список или же чек-боксы, полученная с помощью Monkey и моделью

Приложение, №	Monkey	Модель
5	58%	65%
40	69%	76%
2	61%	63%
6	47%	58%
8	40%	46%
9	53%	53%
10	61%	59%
55	54%	81%
68	53%	67%
67	47%	52%
13	33%	39%
18	30%	36%
65	38%	32%
58	42%	49%
22	42%	42%
25	42%	51%
27	32%	53%
29	37%	43%
31	44%	59%
56	41%	48%
52	42%	42%
49	32%	41%
47	51%	46%
41	40%	47%
34	36%	40%
36	36%	32%

Средний коэффициент покрытия инструментом	44,65%	50,77%

Таблица 4. Максимальное покрытие для каждого приложений, где нет других интерактивных элементов кроме кнопок, полученная с помощью Monkey и моделью

Приложение, №	Monkey	Модель
26	90%	93%
66	64%	85%
12	76%	75%
1	67%	67%
14	60%	62%
16	64%	62%
21	64%	66%
53	73%	75%
44	73%	68%
30	53%	82%
3	80%	58%
19	74%	75%
37	62%	54%
45	71%	61%
Средний коэффициент покрытия инструментом	69,36%	70,21%

По полученным результатам можно сделать вывод, что результаты совпадали в 6 случаях, Monkey показал лучшие результаты в 15 случаях, а разработанная модель – в 47 случаях.

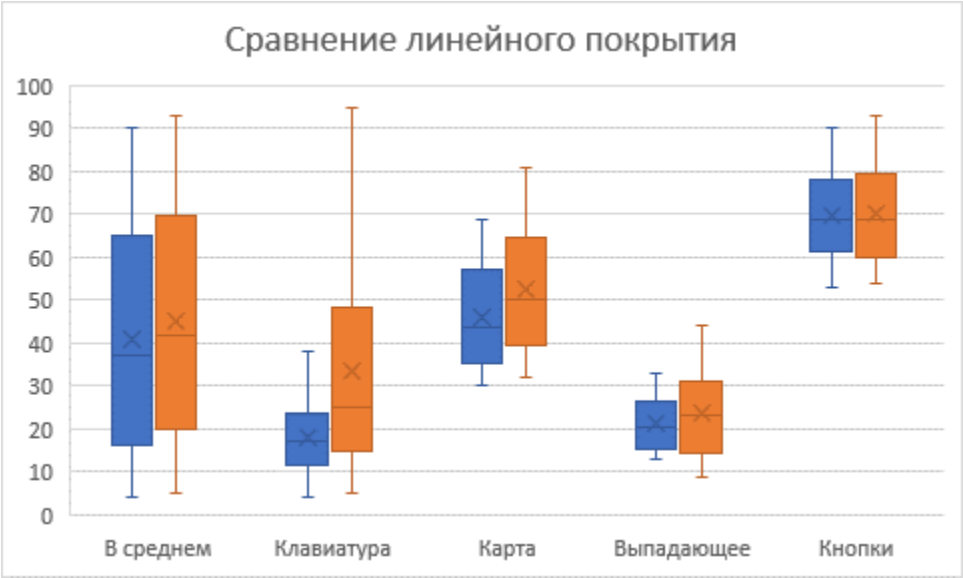

Рис. 1. Сравнение линейного покрытия различных инструментов тестирования для приложений с открытым исходным кодом

Все инструменты использовались с конфигурации по умолчанию. Monkey мог отправлять события ввода с очень высокой скоростью (около 6000 событий/час в эксперименте).

Каждый инструмент тестирования запускался для каждого приложения с открытым исходным кодом на один час. Чтобы рассматривать инструменты и приложения в равных условиях, они запускались на одинаковой версии Android 6.0, так как эта версия поддерживалась всеми приложениями. Для каждого приложения и инструмента записывался окончательный охват и прогрессивный охват после выполнения каждого действия. Данный процесс был повторен три раза и было взято среднее значение в качестве окончательных результатов, представленных в таблице [5]. В среднем модель добилась тестового покрытия строк равного 43,3%, что на 4% выше Monkey.

Модель оказалась лучше в каждой из видов приложений:

1) для приложений, где есть взаимодействие с клавиатурой, модель показала результат 29,14%, что лучше показателя Monkey на 12,57%;

2) для приложений, где есть взаимодействие с картами, модель показала результат 24,71%, что лучше показателя Monkey на 3,35%;

3) для приложений, где есть выпадающий список или же чек-боксы,

модель показала результат 50,77%, что лучше показателя Monkey на 6,12%;

4) для приложений, где нет других интерактивных элементов кроме кнопок, модель показала результат 70,21%, что лучше показателя Monkey на 0,85%.

Подводя итог, можно сказать, что высокое покрытие модели было обусловлено главным образом двумя причинами:

1) модель смогла определить и расставить приоритеты для критически важных элементов пользовательского интерфейса, когда на выбор было множество элементов пользовательского интерфейса.

2) у модели было больше шансов выполнить осмысленную последовательность действий, которая может привести приложение к новым и неизведанным основным функциям.

Эти же данные подтверждаются тем, что для категории, где есть взаимодействие с клавиатурой, разница в % тестового покрытия между Monkey и моделью наибольшая.

Заключение. Случайная стратегия Monkey создавала множество неэффективных и повторяющихся входных событий, что не способствовало быстрому улучшению покрытия в отношении количества событий.

Модель была протестирована и по полученным результатам можно судить, что она превосходит имеющиеся на рынке аналоги.

Библиографический список:

1. Naja Fa., Mansur Sy., Wibawanto Ad. Automated Software Testing on Mobile Applications: A Review with Special Focus on Android Platform // 20th International Conference on Advances in ICT for Emerging Regions. 2020. pp. 4-6.

2. Василенко, Р. И. Автоматизированное тестирование мобильных приложений / Р. И. Василенко, С. А. Белоусова // Инновационные технологии и дидактика в обучении: сборник статей III Международной научно-практической конференции, Краснодар, 29–30 июня 2015 года / Борисова Е.А.. – Краснодар: Издательство Южного Федерального Университета, 2015. – С. 31-34. – EDN

UTWWDN.

3. Михалевская, К. А. Сравнение инструментов для автоматизации тестирования мобильных приложений на ОС Android / К. А. Михалевская, М. А. Сергачева, И. Н. Мерзляков // КОГРАФ - 2020: сборник материалов 30-й Всероссийской научно-практической конференции по графическим информационным технологиям и системам, Нижний Новгород, 13–16 апреля 2020 года. – Нижний Новгород: Нижегородский государственный технический университет им. Р.Е. Алексеева, 2020. – С. 250-255. – DOI 10.46960/43791586_2020_250. – EDN XZGGPU.

4. Сарычева, Ю.Ю., Белов Ю.С. Применение искусственного интеллекта в автоматизированном тестировании GUI // Научные исследования в современном мире. Теория и практика: сборник избранных статей Всероссийской (национальной) научно-практической конференции. 2022. С. 55-56.

5. Ананьев, В. Ю. Автоматизированное тестирование мобильных приложений при помощи nunit и Appium / В. Ю. Ананьев // Инновационные технологии, экономика и менеджмент в промышленности: Сборник научных статей X международной научной конференции, Волгоград, 21–22 октября 2021 года. – Волгоград: Общество с ограниченной ответственностью "КОНВЕРТ", 2021. – С. 136-138. – EDN BBDGXU.