# **5** ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ

Тестирование программного обеспечения (Software Testing) - проверка соответствия между реальным и ожидаемым поведением программы, осуществляемая на конечном наборе тестов, выбранном определенным образом.

В более широком смысле, тестирование – это одна из техник контроля качества, включающая в себя активности по планированию работ (Test Management), проектированию тестов (Test Design), выполнению тестирования (Test Execution) и анализу полученных результатов (Test Analysis).

Верификация (Verification) – это процесс оценки системы или её компонентов с целью определения удовлетворяют ли результаты текущего этапа разработки условиям, сформированным в начале этого этапа. То есть выполняются ли цели, сроки, задачи по разработке проекта, определенные в начале текущей фазы разработки.

Валидация (Validation) – это определение соответствия разрабатываемого ПО ожиданиям и потребностям пользователя, требованиям к системе.

Тестирование проводится для выполнения следующих целей:

* продемонстрировать разработчикам и заказчикам, что программа соответствует требованиям, поставленным в начале разработки продукта;
* выявить ситуации, в которых поведение программы является неправильным, нежелательным или не соответствующим спецификации.

После написания программного кода проводилось полное функциональное тестирование программы.

Для работы приложение в качестве предсказателя необходимо выбирать компьютер с операционной системой семейства Linux.

Тестирование приложения в качестве предсказателя проводилось на следующих устройствах:

* Intel Core i5 7200U 2 ядра по 2,5 ГГц, оперативная память 8 ГБ. Операционная система Arch Linux 4.16;
* Intel Core i5 7200U 2 ядра по 2,5 ГГц, оперативная память 8 ГБ. Операционная система Arch Linux 4.16;
* Intel Core i5 6300HQ 4 ядра по 2,3 ГГц, оперативная память 8 ГБ. Операционная система Arch Linux 4.16.

Для работы приложения в режиме обучения нейронной сети необходимо устройство с операционной системой семейства Linux.

Для тестирования алгоритма обучения модели Child-Sum Tree LSTM использовались следующие устройства:

* Intel Core i5 6200U 2 ядра по 2,3 ГГц, оперативная память 8 ГБ. Операционная система Ubuntu Server 14.10 x64.
* Intel Core i5 3210M 2 ядра по 2,5 ГГц, оперативная память 4 ГБ. Операционная система Ubuntu Server 14.10 x64.
* Intel Core i7 6700HQ 4 ядра по 2,6 ГГц, оперативная память 8 ГБ. Операционная система Ubuntu Server 16.04 x64.

Функциональное тестирование является одним из ключевых видов тестирования, его задачей является установление соответствия разработанного продукта с исходным функциональным требованиям.

В зависимости от степени доступа к коду системы можно выделить два типа функциональных испытаний:

* тестирование black box (черный ящик) – проведение функционального тестирования без доступа к коду системы;
* тестирование white box (белый ящик) – функциональное тестирование с доступом к коду системы.

Тестирование black box проводится без знания внутренних механизмов работы системы и опирается на внешние проявления ее работы. При этом тестировании проверяется поведение программного продукта при различных входных данных и внутреннем состоянии систем.

В случае тестирования white box создаются тест-кейсы, основанные преимущественно на коде системы ПО.

Также существует расширенный тип black box тестирования, включающего в себя изучение кода, – так называемый grey box (серый ящик).

Для тестирования программного продукта были разработаны тест-кейсы. Тест-кейсы для пользователей приведены в таблице 5.1.

Тестирование обучения модели проводилось исходя из того, как влияют гиперпараметры модели на ее производительность. Кроме того, при обучении модели нейронной сети, валидация – это основной прием для контроля качества модели на каждой эпохе обучения. Так же один из способов проверить работу модели – это повторить результаты точности на тренировочном наборе данных, полученные автором модели в исследованиях.

Таблица 5.1 – Тест-кейсы обучения модели

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Предусловие | Содержание теста | Ожидаемый результат | Тест прой-ден |
| 1 | 2 | 3 | 4 | 5 |
| Правдо-подобие точности. | Обучить модель со стандартным набором гиперпара-метров. | Обучить модель, записать результаты точности. | Результаты точности превышают результаты наивного классификатора (значительно больше 60%). | Да. |

*Продолжение таблицы 5.1*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| Дости-жение результа-тов оригина-ла. | Обучить модель с набором гиперпара-метров из оригинального исследования. | Обучить модель, записать результаты точности. | Точность модели в пределах доверительного интервала для точности из исследования (84,9%). | Да. |
| Скорость обучения. | Обучить со стандартными гиперпара-метрами. | Обучить с низкой скоростью обучения (0,01). | Модель обучается медленнее, но потери стабильно падают. | Да. |
| Размер набора. | Обучить со стандартными гиперпара-метрами. | Обучить с высоким размером группы (300). | Обучение займет значительно меньше времени, чем со стандартной скоростью обучения. Точность модели значительно упала. | Да. |

Тестирование предсказателя проще, так как время работы и сложность вычислений в разы проще, чем при обучении модели. Итак, в таблице 5.2 приведены тест-кейсы для предсказателя.

Таблица 5.2 – Тест-кейсы для администратора

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Предусловие | Содержание теста | Ожидаемый результат | Тест прой-ден |
| 1 | 2 | 3 | 4 | 5 |
| Анализ предло-жения с неизвест-ным словом. | Передать в приложение для обработки предложение с одним бессмыслен-ным словом. | Открыть index.html и посмотреть результаты анализа. | Неизвестное слово получило предсказание тональности близкое к нейтральному с нормальным распределением на остальные классы. | Да. |

*Продолжение таблицы 5.2*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| Анализ предло-жения. | Передать в приложение предложение на английском языке. | Открыть index.html и посмотреть результаты анализа. | Предложение успешно проанализиро-вано. Построено синтаксическое дерево. Распределение вероятностей отлично от нормального. | Да. |
| Анализ предло-жения с отсутст-вием семан-тики. | Передать приложению  бессмыслен-ное, лексически правильно составленное, предложение, состоящее целиком из наборов букв: «Сolourless green ideas sleep furiously». | Открыть index.html и посмотреть результаты анализа. | Предложение успешно проанализиро-вано. Построено синтаксическое дерево. Предложение отнесено к классу нейтральной тональности. | Да. |
| Анализ предло-жения с неизвест-ными словами. | Передать приложению предложение, целиком состоящее из бессмыслен-ных наборов букв. | Открыть index.html и посмотреть результаты анализа. | Предложение успешно проанализиро-вано. Построено синтаксическое дерево. Предложение отнесено к классу нейтральной тональности, с нормальным распределением вероятностей других классов. | Да. |

В результате тестирование не было выявлено некорректной работы приложения.