Введение

Одной из актуальных проблем разработки и внедрения программного обеспечения (ПО) является наличие дефектов, то есть ошибок в коде программы, приводящих к снижению качества продукции. Причинами появления дефектов могут стать некачественная организация процесса разработки ПО, недостаточная квалификация и опыт разработчиков, недостаточность ресурсов на разработку. Затраты на выявление и устранение дефектов могут составлять до 80% от общей стоимости ПО [1]. При этом чем раньше будет обнаружен дефект, тем меньше ущерба будет нанесено разработчику и эксплуатанту ПО.

Существует множество техник и технологий для определения дефектов: начиная от ручных средств, заканчивая автоматизированным тестированием ПО. Однако чем больше объем исходного кода, тем больше трудозатрат необходимо для поддержания качества программной системы, при этом ресурсов может не хватать. В данном случае решением могут стать методы машинного обучения, которые на основе ране написанного кода позволят дать вероятностную оценку нахождения дефекта в том или ином месте программы.

Цель данной работы: выявить наиболее эффективные методы машинного обучения для определения дефектов ПО. Для достижения цели необходимо решить следующие **задачи**:

- представить обзор дефектов разрабатываемого ПО и классификацию методов для их обнаружения;
- рассмотреть возможность использования методов машинного обучения в данной сфере;
- сформулировать параметры сравнения методов машинного обучения для обнаружения дефектов ПО;
- провести обзор и сравнение существующих методов машинного обучения для обнаружения дефектов ПО;
- описать формализованную постановку задачи в виде диаграммы в нотации IDEF0.

1 Дефекты разрабатываемого ПО

1.1 Определение понятия дефекта ПО

Согласно стандартному глоссарию терминов и определений, используемых в тестировании ПО, **дефект** – это изъян в компоненте или системе, который может привести компонент или систему к невозможности выполнить требуемую функцию [2]. Другими словами, дефект – это отклонение от первоначальных бизнес-требований, логическая ошибка в исходном коде программы. Дефект не оказывает влияния на функционирование ПО до тех пор, пока он не будет обнаружен при эксплуатации программы. Это может привести к тому, что продукт не будет удовлетворять потребностям пользователя, а также к отказам компонента или системы. Последствия программной ошибки для пользователя могут быть серьезны. Например, дефект может поставить под угрозу бизнес-репутацию, государственную безопасность, бизнес или безопасность пользователей или окружающую среду [3].

Выделяют **три вида** дефектов: программные, технические, архитектурные.

1. *Программные ошибки* – возникают из-за несовершенства исходного кода конечного продукта.
2. *Технические дефекты* – сводятся к доступу тех или иных функций готового решения или его дизайна.
3. *Архитектурные ошибки* – это ошибки, вызванные внешними факторами, которые заранее не были учтены при проектировании решения, вследствие чего приложение демонстрирует результат работы, отличный от ожидаемого [4].

Также с точки зрения степени **влияния на работоспособность** ПО можно выделить пять видов.

1. *Блокирующий дефект (blocker)* – ошибка, которая приводит программу в нерабочее состояние.
2. *Критический дефект (critical)*, приводящий некоторый ключевой функционал в нерабочее состояние? существенное отклонение от бизнес-логики, неправильная реализация требуемых функций и т.д.

3. *Весьма серьезная ошибка (major)*, свидетельствующая об отклонении от бизнес-логики или нарушающая работу программы (не имеет критического воздействия на приложение).
4. *Незначительный дефект (minor)*, не нарушающий функционал тестируемого приложения, но который является несоответствием ожидаемому результату.
5. *Тривиальный дефект (trivial)*, не имеющий влияние на функционал или работу программы, но который может быть обнаружен визуально.

Помимо этого, **по приоритетности исправления** дефекты могут быть с высоким, средним и низким приоритетом.

1. *С высоким приоритетом (high)* – должен быть исправлен как можно быстрее, т.к. критически влияет на работоспособность программы.
2. *Со средним приоритетом (medium)* – дефект должен быть обязательно исправлен, но он не оказывает критическое воздействие на работу.
3. *С низким приоритетом (low)* – ошибка должна быть исправлена, но не имеет критического влияния на программу и устранение может быть отложено [5].

Основными **причинами** возникновения дефектов являются:

- сложность реализации задачи;
- сжатые сроки разработки;
- несовершенство документации;
- изменение требований;
- недостаточная квалификация и опыт разработчиков;
- неправильная организация процесса разработки [4].

1.2 Классификация методов для обнаружения дефектов ПО

Дефекты в программе могут быть обнаружены не сразу и при этом иметь отрицательное влияние на процесс ее использования. При позднем обнаружении дефектов снижаются качество и надежность ПО, увеличиваются затраты на его переработку, проявляются более негативные последствия.

Своевременное выявление и исправление дефектов играют большую роль в жизненном цикле ПО, помогает улучшить качество разрабатываемых систем.

Для выявления дефектов ПО используются различные методы, которые делятся на **три категории**.

1. *Статические методы* – методы, при которых ПО тестируется без какого-либо выполнения программы (системы). При этом программные продукты проверяются вручную или с помощью различных средств автоматизации.
2. *Динамические методы* – в данных методах ПО тестируется путем выполнения программы. С помощью этого метода можно обнаружить различные типы дефектов:
    - функциональные – возникают, когда функции системы работают не в соответствии со спецификацией. Являются критическими, при их наличии программный продукт может работать некорректно и перестать работать;
    - нефункциональные – соответственно затрагивают нефункциональные аспекты приложения, могут повлиять на производительность, удобство использования и т.д.
3. *Операционные методы* – методы, которые производят результат. Дефект идентифицируется или обнаруживается путем проверки – в результате отказа.

Все методы важны и необходимы в процессе управления дефектам. При их объединения достигается наиболее качественный результат, за счет чего повысится производительность ПО. На ранней стадии наиболее эффективными методами являются статические. Они позволяют снизить затраты, необходимые для исправления дефектов, и сводят к минимуму их влияние [6].

Машинное обучение для обнаружения дефектов ПО

Параметры сравнения методов машинного обучения для обнаружения дефектов ПО

Существующие методы машинного обучения для обнаружения дефектов ПО

Формализованная постановка задачи