УДК 004.832.22
DOI: 10.26102/2310-6018/2024.47.4.040

Влияние версии библиотеки TensorFlow на качество генерации кода по изображению

И.В. Никитин✉

Российский экономический университет имени Г.В. Плеханова, Москва, Российская Федерация

Резюме. Данное исследование представляет собой сравнение эффективности обучения моделей, реализующих два разных подхода: усложнение исходной архитектуры нейронной сети либо же сохранение архитектуры при улучшении инструментов, которые используются в основе обучения. Попытки усложнения архитектуры решения по генерации исходного кода на основе изображения приводят к сложно поддерживаемым в будущем решениям. При этом подобные улучшения никак не используют более современные инструменты и библиотеки, на основе которых такие системы построены. Актуальность исследования обусловлена отсутствием попыток использования более современных и актуальных библиотек. В связи с этим, в ходе эксперимента по сравнению показателей моделей трех вариантов систем по генерации исходного кода на основе изображения: оригинальной системы pix2code, ее усложненного варианта и варианта с актуальной версией библиотеки TensorFlow – в процессе их обучения было выявлено, что подходы с усложненной архитектурой и актуальной TensorFlow обладают одинаковыми показателями, более качественными, чем оригинальная pix2code. На основе проведенного эксперимента можно сделать вывод о том, что актуализация библиотеки TensorFlow может дать дополнительный прирост в качестве результатов, которые может предсказывать система по генерации исходного кода на основе изображения.

Ключевые слова: кодогенерация, изображение, машинное обучение, Tensorflow, Keras, предметно-ориентированный язык.

Для цитирования: Никитин И.В. Влияние версии библиотеки TensorFlow на качество генерации кода по изображению. *Моделирование, оптимизация и информационные технологии. Моделирование, оптимизация и информационные технологии.* 2024;12(4). URL: https://moitvivt.ru/ru/journal/pdf?id=1754 DOI: 10.26102/2310-6018/2024.47.4.040

Influence of the TensorFlow library's version on the quality of code generation from an image

I.V. Nikitin✉

Plekhanov Russian University of Economics, Moscow, the Russian Federation

Abstract: This study compares the efficiency of training models that implement two different approaches: complicating the original neural network architecture, or maintaining the architecture while improving the tools used in the training framework. Attempts to complicate the architecture of the solution for generating source code based on an image lead to solutions that might be difficult to support in the future. At the same time, such improvements do not use more modern tools and libraries that such systems are built upon. The relevance of the study is due to the lack of attempts to use more modern and relevant libraries. In this regard, during the experiment to compare the indicators of models of three versions of image-based source code generation systems: the original pix2code system, its complex version, and the version with a modern version of the TensorFlow library - in the process of their training, it was revealed that approaches with a complex architecture and the current TensorFlow have the same indicators, higher quality than the original pix2code. Based on the experiment, we can conclude

that updating the TensorFlow library can provide an additional increase in the quality of results that the image-based source code generation system can predict.

Keywords: code generation, image, machine learning, TensorFlow, Keras, domain-specific language.

For citation: Nikitin I.V. Influence of the TensorFlow library's version on the quality of code generation from an image. *Modeling, Optimization and Information Technology*. 2024;12(4). (In Russ.). URL: https://moitvivt.ru/ru/journal/pdf?id=1754 DOI: 10.26102/2310-6018/2024.47.4.040

Введение

Разработка современного приложения является комплексной задачей, которая требует большого использования ресурсов: как человеческих (время, которое тратится на разработку), так и финансовых (деньги, которые необходимы для разработки и поддержки). Например, создание простого приложения из нескольких экранов может занимать до полугода и стоить значительную сумму денег, которую не каждая компания или индивидуальный предприниматель готовы выделить[1]. В связи с этим, становятся актуальными задачи по оптимизации ресурсов, требуемых для разработки, с целью уменьшения их использования при сохранении качества разрабатываемого продукта. Для решения поставленной задачи в настоящее время активно развивается направление, связанное с использование систем машинного обучения для решения различного рода проблем и упрощения процесса разработки. К примеру, большая языковая модель ChatGPT[2] по текстовому запросу способна предоставить готовый код приложения, а разработка компании Microsoft под название Copilot[3] способна улучшить имеющуюся кодовую базу на основе анализа открытого исходного кода.

Одна из важных составляющих любого программного продукта, с которым предстоит взаимодействовать пользователю – это интерфейс. Взаимодействуя с ним, пользователь может найти в приложении решение той проблемы, которая перед ним стоит. В связи с этим, важным становится вопрос проработки и разработки такого интерфейса. В общем случае, процесс создания интерфейса можно разделить на две части: создание макета дизайна, а после его разработка с использованием исходного кода. Оптимизация данного процесса также может быть сделана за счет моделей нейронных сетей: обученная модель способна предоставить исходный код интерфейса на основе изображения макета. С одной стороны, для решения этой задачи могут быть использованы ранее упомянутые ChatGPT или Copilot, но они имеют ряд ограничений. В частности, ChatGPT предполагает предоставление подробного текстового описания интерфейса, который необходимо создать, а Copilot работает с уже готовым исходным кодом, то есть его необходимо написать.

Существует решение pix2code [1], представленное в 2018 году, и позволяющее на основе изображения макета дизайна приложения предоставить исходный код, который позволяет воспроизвести представленный интерфейс на изображении. В основе pix2code лежит предобученная нейронная сеть, которая реализована с помощью библиотек Keras[4] для подготовки данных и TensorFlow[5] для непосредственного обучения. Идея, лежащая в основе данного продукта, породила череду исследований, пытающихся улучшить механизм получения исходного кода за счет создания более сложной архитектуры, которая позволит уменьшить процент ошибки и повысить качество полученных моделей.

[1] Сколько стоит разработка мобильного приложения в 2024 году. РБК Компании. URL: https://companies.rbc.ru/news/QsGvpEqkZa/skolko-stoit-razrabotka-mobilnogo-prilozheniya-v-2024-godu/ (дата обращения: 20.11.2024).
[2] ChatGPT. URL: https://openai.com/chatgpt/ (дата обращения: 18.11.2024).
[3] Microsoft Copilot: ваш ИИ-помощник. URL: https://copilot.microsoft.com/ (дата обращения: 18.11.2024).
[4] Keras: Deep Learning for humans. URL: https://keras.io/ (дата обращения: 20.11.2024).
[5] TensorFlow. URL: https://www.tensorflow.org/?hl=ru (дата обращения: 25.11.2024).

Последние работы, которые связаны с улучшениями системы pix2code, были опубликованы в 2024 [2]. Интересной особенностью всех этих исследований является то, что они используют версии библиотек Keras и TensorFlow со времен 2018 года. При этом в этих исследованиях не упоминается причина, по которой авторы не пытаются использовать более актуальные версии упомянутых библиотек, чтобы использовать все новые возможности, которые они предоставляют. Предположительно, причин для этого может быть две: либо pix2code слишком сложно актуализировать под использование более новых версий библиотек TensorFlow и Keras, либо обновление библиотек не даст никакого реального прироста качества результатов по сравнению с использование старых версий.

Инструменты, используемые для обучения систем с нейронными сетями, получают бурное развитие в последние годы. С одной стороны, такие инструменты становятся более сложными в своей внутренней реализации, а с другой стороны, системы, которые используют эти инструменты, начинают точнее и быстрее справляться со своими задачами. Например, в обсуждении[6] пользователи и разработчики библиотеки TensorFlow пришли к выводу, что новые версии библиотеки TensorFlow работают быстрее предыдущих. Актуализация используемых инструментов является обычной практикой в случае доработок каких-либо существующих систем для использования всех актуальных возможностей библиотек, однако в случае pix2code этого почему-то не происходило. Актуальность данной работы связана с бурным развитием инструментов обучения нейронных сетей, вызванного широким использованием систем машинного обучения, и отсутствие использования актуальных возможностей этих инструментов в задаче генерации исходного кода на основе изображения. На момент написания данной статьи актуальной версией библиотеки TensorFlow является 2.17 вместо 1.7, которая используется в pix2code и последующих исследованиях.

В рамках данного исследования, сделаны выводы относительного ранее озвученных тезисов, связанных с отсутствием попыток обновления библиотек Keras и TensorFlow: во-первых, определено количество потраченных ресурсов (в первую очередь времени) на актуализацию библиотек, а во-вторых, насколько более актуальные версии упомянутых библиотек эффективнее помогают уменьшить процент ошибок по сравнению с более сложными архитектурными решениями.

Как работает pix2code и его улучшение

Важно отметить, что здесь и далее под исходным кодом, который является выходными данными обученной модели, на самом деле предполагается исходный код на предметно-ориентированном языке, соответствующий определенной части программного кода. Уже после предсказания на предметно-ориентированном языке можно произвести его преобразование к оригинальному исходному коду без использования алгоритмов машинного обучения. Такой подход позволяет работать с абстракциями исходного кода, оставляя за рамками особенности его реализации. Более подробно о том, как будет работать такой подход, и почему он поможет решить задачу генерации исходного кода, можно найти в источнике [3]. В качестве альтернативы, есть системы, одна из которых представлена в работе [8], которые используют для создания исходного кода не предметно-ориентированный язык, а создают сразу напрямую исходный код без промежуточного результата. Однако, результат работы таких систем получается слишком сложным как с точки зрения самого исходного кода, так и с точки зрения его восприятия человеком.

Задача по обучению модели предсказанию исходного кода разделяется на две параллельные подзадачи: обработка изображения и обработка кода, соответствующего

[6] Why is TensorFlow 2 much slower than TensorFlow 1? Issue #33487. tensorflow/tensorflow. GitHub. URL: https://github.com/tensorflow/tensorflow/issues/33487 (дата обращения: 20.11.2024).

этому изображению. Для реализации решения обработки изображения в исходном проекте pix2code используются обычные сверточные сети и архитектура автоэнкодера. Для операции свертки используются фильтры размерностью 32, 64 и 128, а для операции развертывания – наоборот. Одной из особенностей работы с библиотекой Keras является то, что каждый раз операция свертки вызывается дважды с одним и тем же фильтром прежде, чем переходить к следующему шагу. Для обработки исходного кода используются рекуррентные сети, которые представлены как LSTM-ячейки. Исходный код обрабатывается через LSTM-сеть, предварительно пройдя преобразование через механизм быстрого кодирования. Результатом работы обеих подзадач становятся два вектора признаков изображения и исходного кода соответственно. Конкатенация этих векторов на финальном шаге обучения и их обучение с использованием модели Seq2Seq, которая представляет из себя набор последовательных LSTM-сетей, позволяют получить финальное предсказание итогового кода. Когда модель закончила свое обучение, в качестве входных данных в нее передаются изображения. Благодаря обученной модели автоэнкодера, из изображения извлекается вектор признаков того, что находится на изображении, а модель Seq2Seq позволяет преобразовать этот вектор в предсказание исходного кода.

Одним из показателей хорошего качества решения pix2code является то, что оно часто берется в основу последующих улучшений, с целью уменьшения ошибки и увеличения точности предсказания. Существует множество изменений архитектуры, в основе которых лежит pix2code. Так, к примеру, одним из вариантов оптимизации является замена обычных LSTM-ячеек на двунаправленные LSTM-ячейки [4]. Такое улучшение позволяет лучше понимать и запоминать вектор контекста, который передается от одной ячейке другой, что сказалось на результатах работы системы в лучшую сторону. Другой вариант улучшения – это внедрение механизма внимания [5]. Благодаря ему отдельные элементы изображения получают более точные предсказания за счет лучшего понимания нейронной сетью взаимосвязей и ключевых элементов. Еще один вариант улучшения – использование не только двунаправленных LSTM-ячейки, но и другого подхода к обработке изображений с использованием архитектуры ResNet [2]. Такой вариант использует более актуальные архитектуры и подходы, что сказывается на получении более точных результатов работы системы (хотя при это стоит отметить, что в основе все еще лежит TensorFlow 1.7.0). Некоторые исследования пытаются работать не только с тем, чтобы улучшить работу системы pix2code, но и придумать метрики, которые будут более точно определять качество полученной системы на основе именно задачи по созданию исходного кода на основе изображения [7].

Для данного исследования было выбрано улучшение системы pix2code, которое позволяет обрабатывать входные данные не последовательно, а иерархично [6]. Выбор данной системы для сравнения обусловлен тем, что ее исходный код находится в отрытом доступе[7]. Одной из проблем оригинальной системы, как отмечают авторы статьи [6], являются те самые LSTM-ячейки. Из-за архитектуры самих ячеек и сети на их основе не получится хранить большой объем информации. Это может привести к проблеме неправильного результата при попытке обработки изображения, содержащего большое количество элементов, так как сети может не хватить памяти, чтобы их все запомнить и дать предсказание об исходном коде. Другой потенциальной проблемой может являться то, что исходный код является иерархичной структурой. Поэтому даже если системе хватит памяти, чтобы удержать всю информацию об изображении, ее может не хватить для построения правильной иерархии в итоговом предсказании. Для решения указанной проблемы, авторами статьи было предложено решение по иерархичной обработке

[7] ZhihaoZhu/Auto-GUI-Code-Generation. GitHub. URL: https://github.com/ZhihaoZhu/Auto-GUI-Code-Generation (дата обращения: 13.11.2024).

изображения. Вместо того, чтобы сразу обрабатывать все исходное изображение, можно его обработать таким образом, чтобы сначала выделить верхнеуровневую структуру того, что представлено на изображении, а уже после этого обработать каждый из блоков. При этом обработка на каждом из этапов схожа с тем, как работает система pix2code, когда сначала формируется вектор признаков элементов на изображении, после чего они преобразовываются в исходный код, который затем собирается в уже итоговый результат. Эксперименты, представленные в статье, показали, что такая система обладает большей точностью по сравнению с pix2code на одинаковом наборе данных, что говорит о том, что архитектура с иерархичной обработкой позволяет достичь цели по улучшению результатов. Также стоит отметить, что данная система также реализована с использованием TensorFlow 1.7.0.

Изменения в библиотеках TensorFlow и Keras

Если обратиться к исходному коду обоих систем, то можно обратить внимание, что они обе используют уже устаревшие версии библиотек Keras 2.x и TensorFlow 1.x. За прошедшее время обе библиотеки были переработаны и представлены в новых версиях, в которых улучшены способы работы с данными, их обучение, а также изменены некоторые методы и функции, предоставляемые разработчику.

Библиотека TensorFlow была представлена в 2015 году. Ее основная цель – предоставить интерфейс и методы для возможностей обучения нейронных сетей. Основная идея, которая лежит в основе библиотеки – это построение и работа с вычислительным графом. Несмотря на то, что поддерживать граф довольно сложно с точки зрения исходного кода, такой подход обладает рядом плюсов: часть операций может вызываться параллельно и независимо друг от друга. Благодаря этому открываются возможности по оптимизации производительности работы библиотеки через запуск нескольких задач одновременно.

В 2019 году была выпущена новая версия 2.0, которая и продолжает на момент написания данной статьи получать различные обновления, улучшающие как производительность, так и функционал. Эта мажорная версия получила большое количество обновлений, из которых можно выделить следующие вещи: нативная поддержка библиотеки Keras, которая теперь является частью TensorFlow; низкоуровневая оптимизация исходного кода для ускорения получения результатов; а также изменения потока выполнения программы. Если раньше программу и данные сначала надо было полностью описать, а после начать ее исполнять, то теперь возможно последовательное исполнение с проверкой результатов на каждом из этапов.

Основное преимущество подхода с просто переходом на TensorFlow 2.x по сравнению с изменением архитектуры приложения заключается в том, что в системе остается оригинальная архитектура, которая не претерпевает каких-либо изменений. При этом не стоит забывать, что TensorFlow продолжает развиваться: появление новых версий связано как с исправлением известных проблем, так и с оптимизацией и расширением функционала. Это значит, что использование актуальных версий позволит в полной мере раскрывать возможности библиотеки. Однако при этом, процесс перехода на новые версии библиотек может быть настолько сложным, что система без подготовительных работ не сможет быть переведена на TensorFlow 2.x.

Эксперимент

В рамках эксперимента были проверены ранее озвученные гипотезы: о слишком больших ресурсных затратах на актуализацию TensorFlow и Keras, а также о повышении качества полученных результатов после обновления. Для этого необходимо, во-первых, внести изменения в исходный код приложения pix2code для его работы с новыми версиями

библиотек, а во-вторых, на полученном коде произвести обучение и сравнить полученные результаты.

Для актуализации исходного кода была выбрана версия TensorFlow 2.17 как наиболее актуальная на момент проведения эксперимента. Успешным переписыванием кода в данном случае будем считать состояние, когда система без ошибок подготавливает данные для обучения, а после способна обучить как минимум одну эпоху. При такой конфигурации впоследствии можно начинать обучение на большем количестве эпох. В итоге, на актуализацию под новые версии библиотек потребовалось время, равное одному 8-ми часовому рабочему дню одного разработчика. В качестве примера для сравнения, для последующего обучения 50-ти эпохам одного варианта системы ушло порядка 50-ти часов (а итоговое время на все эксперименты было еще больше). В результате, можно говорить о том, что актуализация TensorFlow не потребовала больших временных затрат, поэтому гипотеза о том, что этот процесс слишком сложный, оказалась неверна. При этом в работе [6] нет информации о том, сколько потребовалось времени на разработку системы.

Для проверки гипотезы о том, что актуализированные версии библиотек TensorFlow и Keras не способны дать какого-либо прироста в качестве полученных результатов, был проведен эксперимент, в ходе которого произошло сравнение результатов трех программ: оригинальная pix2code с использованием библиотеки TensorFlow версии 1.7.0; иерархичная pix2code, представленная в статье [6] с использованием библиотеки TensorFlow версии 1.7.0; pix2code, использующая TensorFlow версии 2.17. В ходе этого эксперимента сравнивались показатели Accuracy и Loss в процессе обучения всех трех систем на протяжении 50 эпох для трех платформ: Web, Android, iOS. Обучение происходило на процессоре M1 от компании Apple. В качестве данных используется набор данных из библиотеки PixCo, представленный в оригинальной статье pix2code[8]. Результаты экспериментов для платформы Web представлены на Рисунках 1–2. На Рисунках 3–4 представлены результаты для платформы iOS, а на Рисунках 5–6 показаны результаты экспериментов для платформы Android.

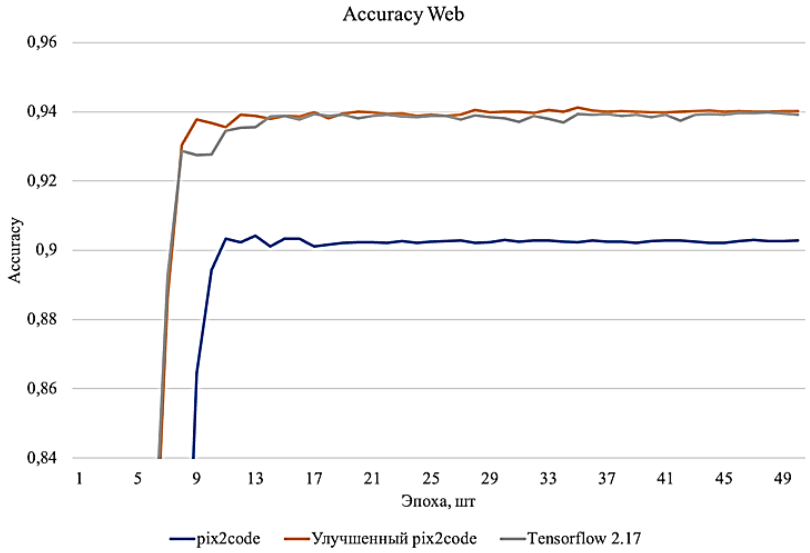

Рисунок 1 – Значение параметра Accuracy во время обучения генерации исходного кода под платформу Web

Figure 1 – Value of the Accuracy parameter during training a generation of source code for the Web platform

[8] pix2code/datasets at master. tonybeltramelli/pix2code. GitHub. URL: https://github.com/tonybeltramelli/pix2code/tree/master/datasets (дата обращения: 13.11.2024).

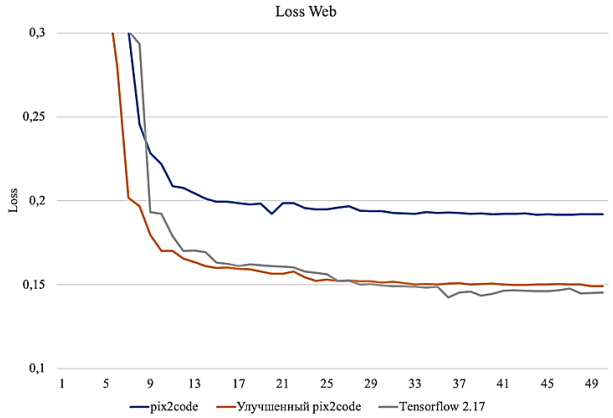

Рисунок 2 – Значение параметра Loss во время обучения генерации исходного кода под платформу Web
Figure 2 – Value of the Loss parameter during training a generation of source code for the Web platform

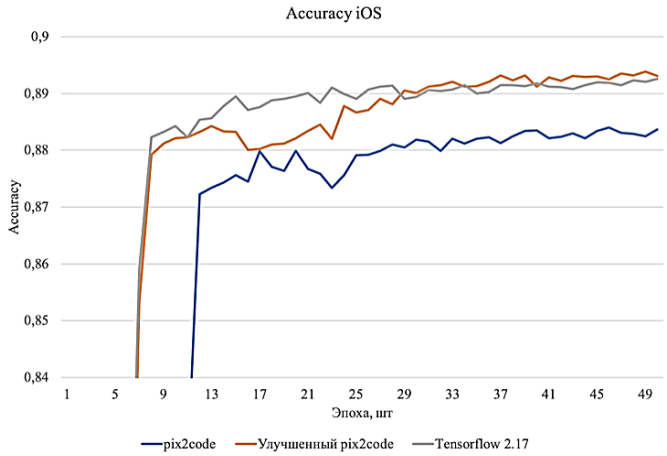

Рисунок 3 – Значение параметра Accuracy во время обучения генерации исходного кода под платформу iOS
Figure 3 – Value of the Accuracy parameter during training a generation of source code for the iOS platform

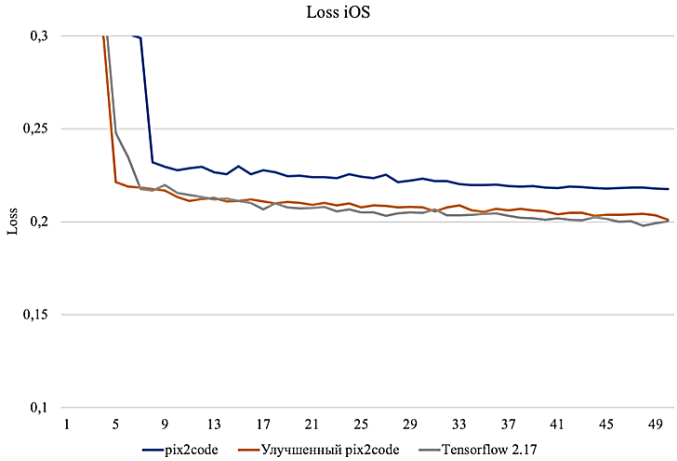

Рисунок 4 – Значение параметра Loss во время обучения генерации исходного кода под платформу Web
Figure 4 – Value of the Loss parameter during training a generation of source code for the Web platform

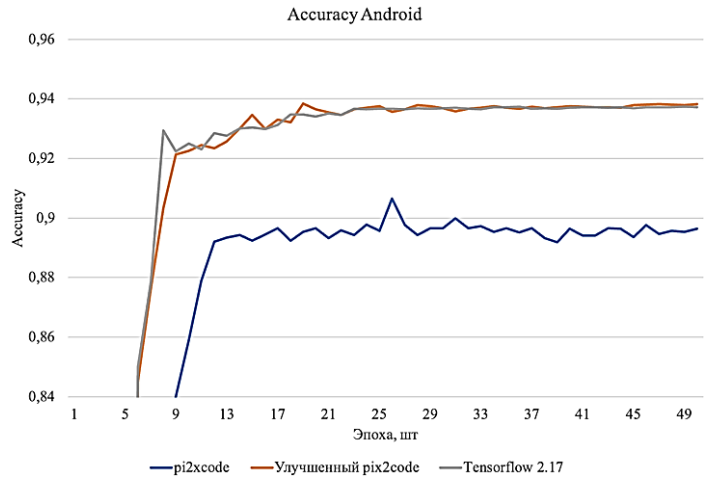

Рисунок 5 – Значение параметра Accuracy во время обучения генерации исходного кода под платформу Android

Figure 5 – Value of the Accuracy parameter during training a generation of source code for the Android platform

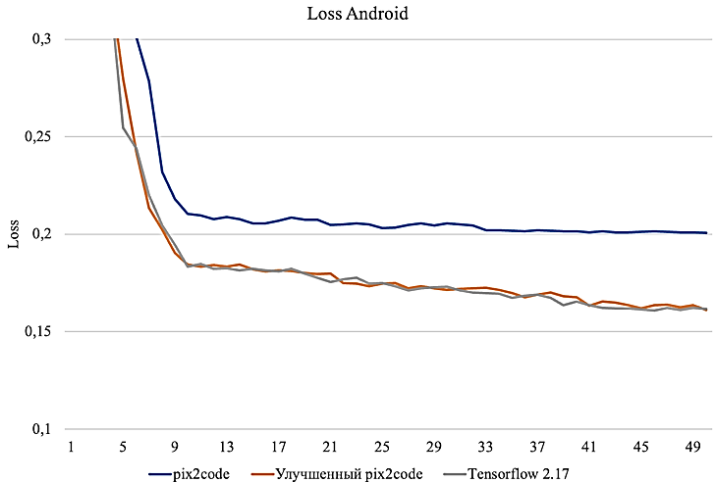

Рисунок 6 – Значение параметра Loss во время обучения генерации исходного кода под платформу Android

Figure 6 – Value of the Loss parameter during training a generation of source code for the Android platform

Как можно увидеть, сам по себе проект pix2code показывает результаты хуже, чем два других варианта моделей, однако со своей основной задачей он все-таки справляется и способен предоставить предсказание. Если же сравнивать варианты улучшенной архитектуры и замены библиотеки TensorFlow 1.x на TensorFlow 2.x, то результаты таких систем уже будут схожими между собой. Можно обратить внимание, что есть определенная разница в самих значениях параметров обучения между платформами (Web результаты лучше, а iOS, наоборот, хуже в сравнении трех платформ). Однако при этом на общий тренд это никак не влияет – оригинальный pix2code все еще проигрывает, а оба улучшенных варианта лучше в плане своих результатов.

Из этого эксперимента можно сделать вывод, что на тех открытых данных, которые можно использовать в обучении, вариант с переписыванием старых систем, использующих TensorFlow 1.x, на более актуальный TensorFlow 2.x достаточно для того, чтобы получить прирост в точности предсказания, сравнимый с использованием более

комплексных архитектурных решений, а значит гипотеза о том, что обновление библиотек не даст никакого прироста в качестве, не верна.

Таким образом, в ходе проведенных экспериментов, можно сделать вывод о том, что изначально выдвинутые гипотезы о причинах, по которым до этого не происходило обновление библиотек, не верны. Также в ходе экспериментов была получена система, которая использует в своей основе актуальные версии библиотек TensorFlow и Keras, которую можно продолжить развивать с целью получения более качественных результатов, не забывая при этом поддерживать версии библиотек в актуальном состоянии. Исходный код полученной системы с обновленными библиотеками можно найти по источнику[9].

Заключение

В ходе исследования были выдвинуты гипотезы о том, почему исследования, связанные с улучшением системы pix2code по генерации исходного кода, используют старые версии библиотек TensorFlow и Keras: это слишком долго и сложно или это не даст какого-то прироста метрик качества.

Результаты эксперимента показали, что, во-первых, для актуализации исходного кода необходимо время порядка 8-ми часового рабочего дня одного инженера, что является сравнительно небольшим показателем, например, относительно времени обучения системы. Во-вторых, вариант pix2code с более сложной архитектурой и вариант с актуализацией кодовой базы под версию TensorFlow 2.x схожи в своих результатах, и оба показали результаты лучше оригинальной системы pix2code. Таким образом, изначальные предположения о причинах неиспользования новых версий библиотек оказались неверны.

В будущих исследованиях имеет смысл опираться уже на ту версию pix2code, которая использует версию библиотеки TensorFlow 2.17. Кроме того планируется рассмотреть другие улучшения, которые могут помочь дать более точные предсказания исходного кода. Например, использование архитектуры трансформера для работы с текстовой информацией как вариант обработки текстовой информации (эффективность его использования для обработки текстовой информации можно найти в оригинальной статье [10]) и архитектуру RetinaNet для улучшения части, связанной с распознаванием изображения (архитектурные особенности и эффективность работы такой архитектуры можно найти в статье [9]). Вероятно, такая комбинация улучшений может дать прирост в качестве результатов лучше, чем был получен до этого.

СПИСОК ИСТОЧНИКОВ / REFERENCES

1. Beltramelli T. pix2code: Generating Code from a Graphical User Interface Screenshot. In: *EICS '18: Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems, 19–22 June 2018, Paris, France.* New York: Association for Computing Machinery; 2018. https://doi.org/10.1145/3220134.3220135
2. Zou D., Wu G. Automatic Code Generation for Android Applications Based on Improved Pix2code. *Journal of Artificial Intelligence and Technology.* 2024;4(4):325–331. https://doi.org/10.37965/jait.2024.0515
3. Balog M., Gaunt A.L., Brockschmidt M., Nowozin S., Tarlow D. DeepCoder: Learning to Write Programs. In: *5th International Conference on Learning Representations, ICLR*

[9] IlyaNikk/html-code-generation: Using the AI model to generate HTML Code by giving a UI mockup image. GitHub. URL: https://github.com/IlyaNikk/html-code-generation (дата обращения: 13.11.2024).

2017: Proceedings, 24–26 April 2017, Toulon, France. 2017. https://doi.org/10.48550/arXiv.1611.01989

4. Liu Y., Hu Q., Shu K. Improving pix2code based BI-directional LSTM. In: *2018 IEEE International Conference on Automation, Electronics and Electrical Engineering (AUTEEE), 16–18 November 2018, Shenyang, China.* IEEE; 2019. pp. 220–223. https://doi.org/10.1109/AUTEEE.2018.8720784
5. Chen W.-Y., Podstreleny P., Cheng W.-H., Chen Y.-Y., Hua K.-L. Code generation from a graphical user interface via attention-based encoder-decoder model. *Multimedia Systems.* 2022;28(1):121–130. https://doi.org/10.1007/s00530-021-00804-7
6. Zhu Z., Xue Z., Yuan Z. Automatic Graphics Program Generation Using Attention-Based Hierarchical Decoder. In: *Computer Vision – ACCV 2018: 14th Asian Conference on Computer Vision: Revised Selected Papers: Part VI, 02–06 December 2018, Perth, Australia.* Cham: Springer; 2019. pp. 181–196. https://doi.org/10.1007/978-3-030-20876-9_12
7. Yao X., Yap M.H., Zhang Y. Towards a Deep Learning Approach for Automatic GUI Layout Generation. In: *Proceedings of International Conference on Computing and Communication Networks, ICCCN 2021, 19–20 November 2021, Manchester, United Kingdom.* Singapore: Springer; 2022. pp. 19–27. https://doi.org/10.1007/978-981-19-0604-6_2
8. Nguyen T.A., Csallne C. Reverse Engineering Mobile Application User Interfaces with REMAUI (T). In: *2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE), 09–13 November 2015, Lincoln, USA.* IEEE; 2016. pp. 248–259. https://doi.org/10.1109/ASE.2015.32
9. Lin T.-Y., Goyal P., Girshick R., He K., Dollár P. Focal Loss for Dense Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 2020;42(2):318–327. https://doi.org/10.1109/TPAMI.2018.2858826
10. Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A.N., Kaiser Ł., Polosukhin I. Attention Is All You Need. In: *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems, 04–09 December 2017, Long Beach, USA.* New York: Curran Associates Inc.; 2017. pp. 6000–6010.

ИНФОРМАЦИЯ ОБ АВТОРАХ / INFORMATION ABOUT THE AUTHORS

Никитин Илья Владимирович, аспирант Российский экономический университет имени Г.В. Плеханова, Москва, Российская Федерация.
e-mail: vic096@yandex.ru

Ilya V. Nikitin, Graduate student, Plekhanov Russian University of Economics, Moscow, the Russian Federation.

Статья поступила в редакцию 26.11.2024; одобрена после рецензирования 19.12.2024; принята к публикации 24.12.2024.

The article was submitted 26.11.2024; approved after reviewing 19.12.2024; accepted for publication 24.12.2024.