УДК 004.832.22
DOI: 10.26102/2310-6018/2025.49.2.002

Использование архитектур ResNet и Трансформеров в задаче генерации исходного кода на основе изображения

И.В. Никитин✉

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

Резюме. В статье рассматриваются различные способы оптимизации системы, разработанной для генерации исходного кода на основе изображения. Сама система состоит из двух частей: автоэнкодера для обработки изображений и выделения из них необходимых признаков, и обработки текста с использованием LSTM блоков. В последнее время вышло много новых подходов к решению задач как улучшения показателей обработки изображения, так и обработки и предсказания текста. В рамках данного исследования были выбраны архитектуры ResNet для улучшения части, связанной с обработкой изображения, и архитектура Трансформера для улучшения части, связанной с предсказанием текста. В рамках экспериментов было проведено сравнение показателей систем, состоящих из различных комбинаций архитектурных решений исходной системы, ResNet архитектуры и Трансформеров, сделан вывод о качестве предсказания на основе показателей метрик BLEU, chrF++, а также выполнения функциональных тестов. В ходе проведенных экспериментов был сделан вывод о том, что комбинация архитектур ResNet и Трансформеров показывает наилучший результат в задаче генерации исходного кода на основе изображения, но также эта комбинация требует наибольшего времени для своего обучения.

***Ключевые слова*:** кодогенерация, изображение, машинное обучение, ResNet, Трансформеры.

Для цитирования: Никитин И.В. Использование архитектур ResNet и Трансформеров в задаче генерации исходного кода на основе изображения. *Моделирование, оптимизация и информационные технологии*. 2025;13(2). URL: https://moitvivt.ru/ru/journal/pdf?id=1863 DOI: 10.26102/2310-6018/2025.49.2.002

Using ResNet and Transformer architectures in the problem of source code generation from an image

I.V. Nikitin✉

Plekhanov Russian University of Economics, Moscow, the Russian Federation

Abstract. This study examines different ways to optimize a system designed to generate source code from an image. The system itself consists of two parts: an autoencoder for processing images and extracting the necessary features from them, and text processing using LSTM blocks. Recently, many new approaches have been released to solve problems of both improving image processing performance and text processing and prediction. In this study, ResNet architectures were chosen to improve the image processing part and Transformer architecture to improve the text prediction part. As part of the experiments, a comparison was made of the performance of systems consisting of various combinations of architectural solutions of the original system, ResNet architecture and transformers, and a conclusion was made about the quality of prediction based on the performance of the BLEU, chrF++ metrics, as well as the execution of functional tests. The experiments showed that the combination of ResNet and Transformer architectures shows the best result in the task of generating source code from an image, but this combination also requires the longest time for its training.

Keywords: code generation, image, machine learning, ResNet, Transformers.

For citation: Nikitin I.V. Using ResNet and Transformers architectures in the problem of source code generation from an image. *Modeling, Optimization and Information Technology*. 2025;13(2). (In Russ.). URL: https://moitvivt.ru/ru/journal/pdf?id=1863 DOI: 10.26102/2310-6018/2025.49.2.002

Введение

Использование систем машинного обучения плотно входит в нашу повседневную жизнь. Многие сферы пытаются интегрировать их в свои процессы, как, например, в медицине, транспортной безопасности и других. В повседневной жизни все чаще мы обращаемся к различным чат-ботам, способным по запросу предоставить необходимый результат. Самый известный пример – ChatGPT. Однако, последнее время появляется много аналогов, которые работают схожим образом, но построены на несколько других архитектурах, что может сказаться на результатах. Например, недавно появившиеся чат-боты DeepSeek или Grok являются прямыми конкурентами ChatGPT.

Помимо того, что на рынке появляется множество различных чат-ботов, способных решать повседневные задачи, такой бурный рост решений с использованием систем машинного обучения также приводит к тому, что активно развиваются технологии и подходы, которые используются в их основе. Так, например, благодаря архитектуре Трансформер-а стало возможным создание больших языковых моделей, лежащих в основе современных чат-ботов.

Одна из задач, которую способны решить системы машинного обучения – это генерация исходного кода на основе изображения. В общем виде процесс работы такой системы можно описать как «понять, что на изображении, и предоставить исходный код в виде текста». Именно это и реализовано в системе pix2code в рамках работы [1]. Она представляет собой рабочую систему, способную на основе изображения предоставить исходный код того, что представлено на изображении. Делается это с помощью преобразования изображения через сверточные сети, а после с использованием LSTM-сети предсказывается исходный код.

Хотя эта система и способна выполнять поставленную перед ней задачу, существует ряд исследований, в которых происходят доработки данной системы, позволяющие повысить качество полученного кода. Так, например, в исследовании [2] процесс работы с изображением сводится к разбиению изображения на маленькие части, работы с ними, а после агрегации полученных кусочков в итоговый ответ. В другом исследовании [3] более точечно меняется архитектура LSTM на BLSTM, что позволило несколько улучшить показатели. Еще один пример – это исследование [4]. В нем авторы уже делают замену как части, связанной с LSTM на BLSTM, так и части по распознаванию изображения на архитектуру ResNet.

Ранее мной в рамках исследования [5] уже был проведен эксперимент по замене библиотеки TensorFlow, лежащей в основе pix2code на актуальную версию 2.x, что также показало рост качества относительно оригинальной pix2code. В другом исследовании [6] мной была подготовлена система по оценке качества полученного исходного кода за счет выполнения этого самого кода: получение изображения, выполнение предсказанного исходного кода и его дальнейшее сравнение с эталонным изображением. Однако, по сути, в этих исследования все еще использовались архитектурные подходы, которые были реализованы в оригинальной системе pix2code. За последнее время появилось много других архитектурных решений, часть из которых была реализована в других работах, а часть еще нет. Некоторые из актуальных подходов потенциально помогут повысить качество полученного кода. Актуальность данной работы заключается в повышении качества предсказанного исходного кода за счет применения более

комплексных и актуальных архитектурных подходов для задачи генерации исходного кода на основе изображения.

В рамках данного исследования проведен эксперимент, в ходе которого архитектура системы pix2code была изменена за счет использования архитектуры ResNet для определения параметров изображения, а также использования подхода Трансформеров для генерации самого кода. По итогам выполненных переработок исходного кода системы проведено обучение, результаты которого сравнивались с системой, полученной в работе [6].

Обзор архитектур ResNet и Трансформеров

Как ранее было отмечено, система pix2code состоит из двух подсистем: одна для обработки изображений и выделения из них параметров или блоков того, что есть на изображении, а вторая – для непосредственного предсказания исходного кода на основе выделенных из изображения признаков. С точки зрения реализации обработка изображения сделана с использованием стандартных функций TensorFlow для работы с изображением. Таким образом реализована архитектура сверточной сети для обработки изображений 256 на 256 пикселей. Предсказание исходного кода, в свою очередь, реализовано через две LSTM-сети, которые вместе образуют архитектуру Seq2Seq моделей. Также стоит отметить, что обработка изображений реализована как автоэнкодер. Это позволяет системе обучаться без заданных параметров, а узнавать про них по ходу своего обучения. Обе эти подсистемы могут быть модифицированы за счет использования архитектур ResNet и Трансформеров.

ResNet – это архитектура сети, созданная специально для оптимального обучения для задачи классификации, представленной в исследовании [7]. В основе такой сети лежат соединения быстрого доступа. Благодаря таким соединениям удается избежать проблемы быстрого роста или затухания функции во время обучения. Кроме того, эксперименты показали, что чем больше слоев в сверточной сети, тем на самом деле больше процент ошибок. Архитектура ResNet не обладает такой проблемой как раз за счет наличия блоков быстрого доступа, и при одинаковом количестве слоев с обычной сетью способна показать результаты лучше (а также выполнять задачу оптимальнее за счет меньшего количества операций). Так, в исследовании [8] как раз приводятся рассуждения о том, как именно можно решить проблему затухающего градиента или резко возрастающего, и подход с блоком быстрого соединения позволит решить эту проблему. Как отмечалось ранее, идея использовать сеть ResNet в системе pix2code была предложена в статье [4]. Хотя помимо использования ResNet в этом же исследовании было предложено использовать и BLSTM, совместно они показали хороший результат в процессе обучения, хотя для этого использовались только метрики Loss и Accuracy, которые, как я отмечал в статье [6], плохо отражают реальные результаты обучения. К сожалению, проверить результат работы системы из исследования [4] на других метриках возможности нет в силу отсутствия исходного кода в открытом доступе. Стоит также отметить, что архитектура ResNet используется в первую очередь именно для задачи классификации, поэтому для ее использования в качества автоэнкодера требуются доработки, которые были реализованы для проведения экспериментов.

Трансформеры – это архитектура нейронных сетей, впервые представленная в исследовании [9]. Ее основная задача – это различные виды работы с текстовой информацией, например, ее классификация или создание новой текстовой информации. В последнее время эту архитектуру можно видеть в реализации различных чатов-ботов наподобие ChatGPT. В своей основе Трансформеры представляют собой автоэнкодер, в некоторой степени похожий на тот, что используется в основе архитектуры модели

Seq2Seq, но с изменениями. Во-первых, в отличие от того же Seq2Seq, где каждое слово или символ обрабатывается последовательно, Трансформеры обрабатывают информацию параллельно. Это позволяет ускорить процесс работы системы, а также позволяет обрабатывать большее количество данных. Во-вторых, Трансформеры используют механизм внимания. Благодаря ему, получается лучше провести, например, синтаксический разбор исходного текста с целью понять из чего состоит данное предложение, а также понять, о чем в целом это предложение для того, чтобы предсказание системы имело смысл. Эффективность использования механизма внимания для улучшения показателей предсказания кода можно увидеть в [10]. В данном исследовании используется последовательная генерация слов с использованием ячеек GRU вместо LSTM, а также внедрен механизм внимания для приоритезации выбора того или иного слова. Однако эта реализация не совпадает с тем, что закладывается в Трансформеры.

Эксперименты

Рассмотренные ранее архитектуры должны показать результаты лучше, чем оригинальные архитектурные решения, используемые в pix2code. Для того, чтобы проверить гипотезу о том, что это действительно так, был проведен ряд экспериментов с последующим сравнением результатов.

Библиотека TensorFlow предоставляет три варианта реализации архитектуры ResNet: ResNet50, ResNet101, ResNet152. Отличаются они количеством слоев в реализации (50, 101 и 152 соответственно): чем больше слоев, тем большее количество операций делается. Для проведения экспериментов была выбрана сеть ResNet50 как сеть, которая с одной стороны позволит проверить гипотезу о том, что данная сеть в целом позволяет получить результат лучше, чем в исходной системе, а с другой – ее обучение не будет слишком долгим и затратным с вычислительной точки зрения.

За основу была взята система pix2code, представленная в исследовании [6] (далее – исходная система). Она использует актуальную версию библиотеки TensorFlow 2.17, а также имеет механизмы по оценке качества полученного кода: Bleu и функциональные тесты. Кроме того, был еще добавлен расчет метрики chrF++ из исследования [11], как еще одна метрика для оценки качества, а также более современная метрика по расчету соответствия n-грамм.

Для каждого из экспериментов обучение проводилось на протяжении 50 эпох на процессоре M1. В качестве набора данных использовался набор из оригинальной системы pix2code. Для сравнения результатов во всех графиках использовалась одна и та же пара изображения и кода, соответствующего изображению, которые использовались как входные данные в исходную систему. Исходный код программы можно найти по источнику[1].

В рамках первого эксперимента исходная система сравнивается с измененной, в рамках которой оригинальная реализация автоэнкодера для обработки изображений заменяется на вариант, реализованный с использованием ResNet50. Предсказание исходного кода остается без изменений и происходит с использованием LSTM-сетей. Результаты представлены на Рисунках 1 и 2.

[1] IlyaNikk / html-code-generation. GitHub. URL: https://github.com/IlyaNikk/html-code-generation (дата обращения: 18.03.2025).

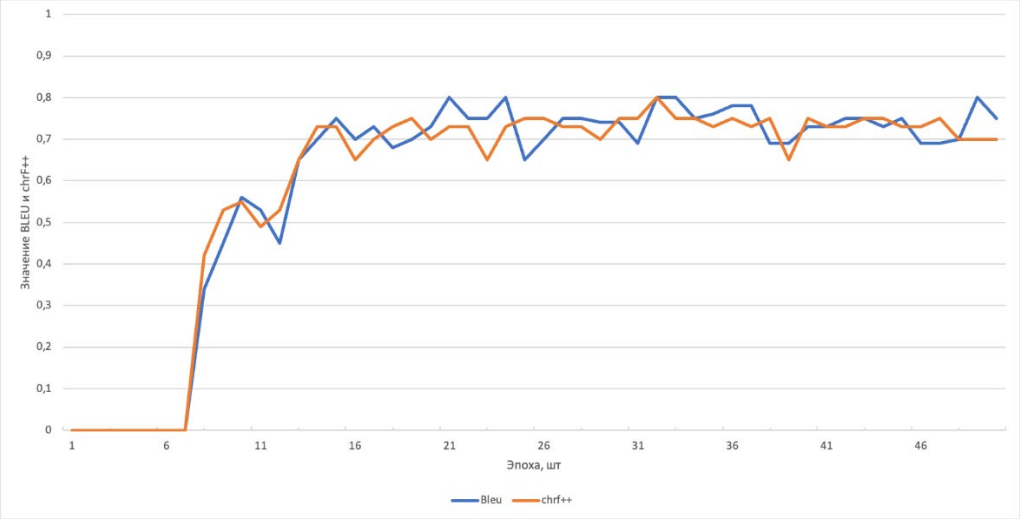

Рисунок 1 – Изменения значения метрики BLEU и chrF++ в процессе обучения с использованием архитектуры ResNet
Figure 1 – Changes in BLEU and chrF++ metric values during training using ResNet architecture

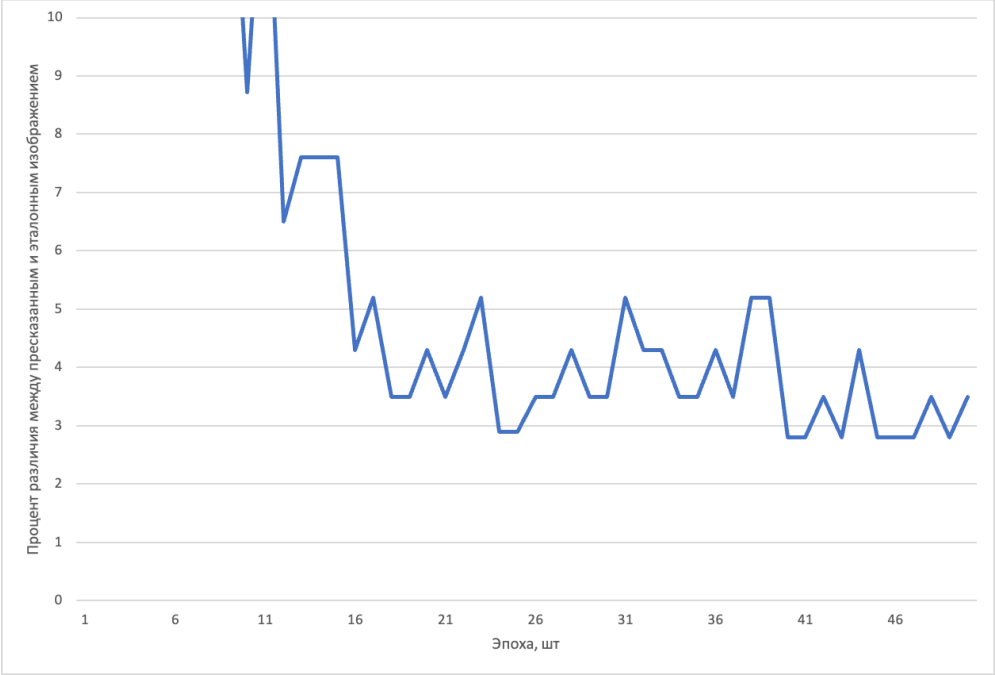

Рисунок 2 – Изменения результата выполнения функциональных в процессе обучения с использованием архитектуры ResNet
Figure 2 – Changes of functional test's result during training using ResNet architecture

Как видно на основе метрик, результаты оказались лучше, чем у исходной системы, однако нельзя сказать, что они стали значительно лучше и виден значительный скачок в качестве. Также, можно обратить, что метрики BLEU и chrF++ стали более стабильными: по сравнению с исходной системой, в них стало меньше выбросов.

В рамках второго эксперимента исходная система сравнивается с измененной, в рамках которой сети LSTM заменяются на реализацию через Трансформер для изменения способа предсказания исходного кода, а обработка изображений остается прежней. Результаты представлены на Рисунках 3 и 4.

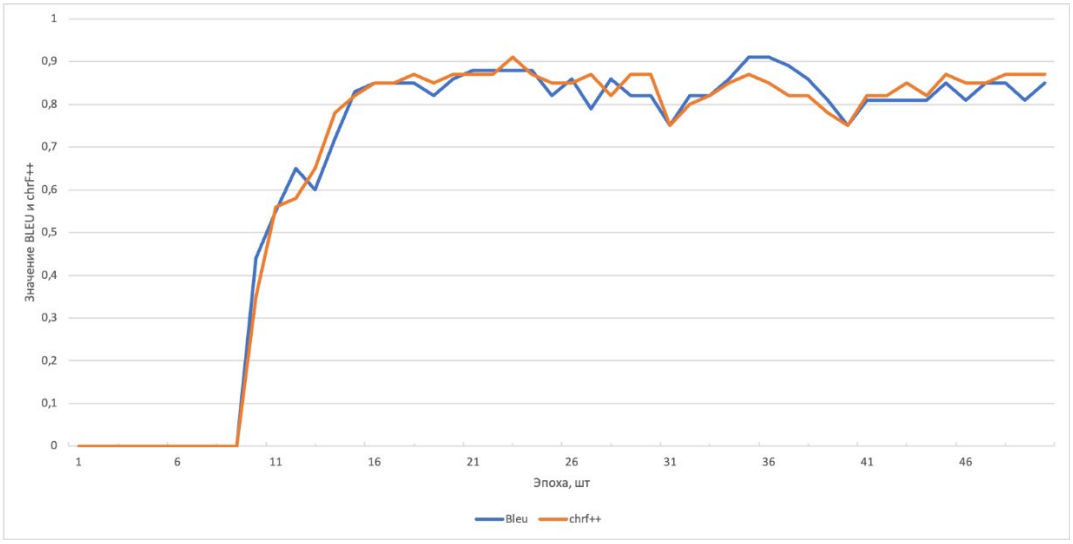

Рисунок 3 – Изменения значения метрики BLEU и chrF++ в процессе обучения с использованием архитектуры Трансформеров
Figure 3 – Changes in BLEU and chrF++ metrics during training using the Transformers architecture

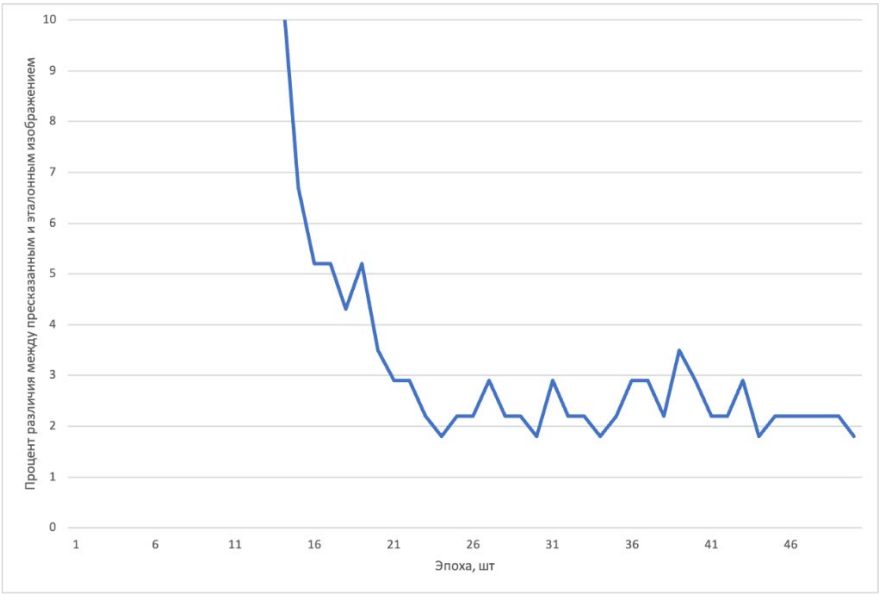

Рисунок 4 – Изменения результата выполнения функциональных тестов в процессе обучения с использованием архитектуры Трансформеров
Figure 4 – Changes of functional test's result during training using Transformer architecture

Как можно заметить, результат в данном эксперименте оказался также лучше, чем в исходной системе, но помимо этого, он оказался также лучше и того результата, что был получен в ходе первого эксперимента. Однако, стоит отметить, что в данном случае системе потребовалось больше эпох на то, чтобы начать получать результат предсказания: вместо 6 в исходной системе и системе из первого эксперимента понадобилось 9 эпох обучения.

В рамках третьего эксперимента исходная система сравнивается с новой, в которой используется как архитектура ResNet для обработки изображений, так и Трансформеры для предсказания текстов. Результаты экспериментов представлены на Рисунках 5 и 6.

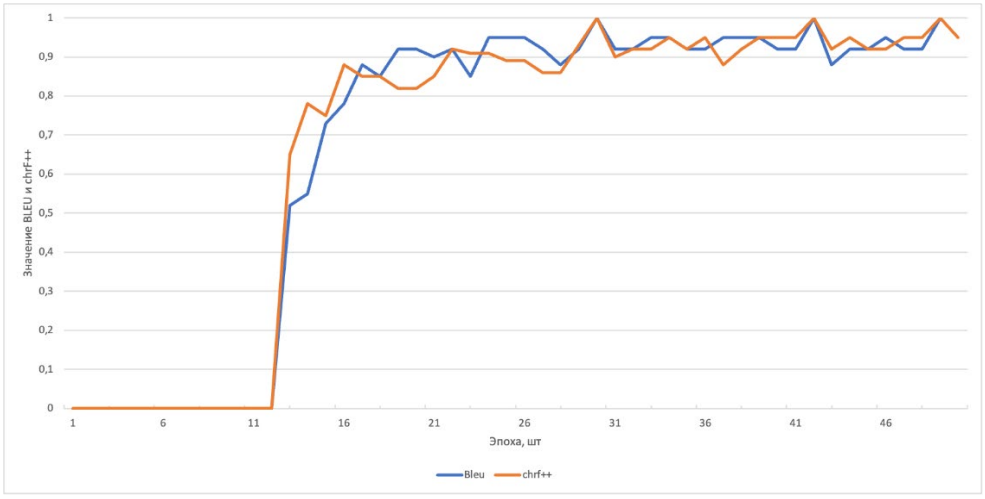

Рисунок 5 – Изменения значения метрики BLEU и chrF++ в процессе обучения с использованием архитектур ResNet и Трансформеров
Figure 5 – Changes in BLEU and chrF++ metric values during training using ResNet and Transformers architectures

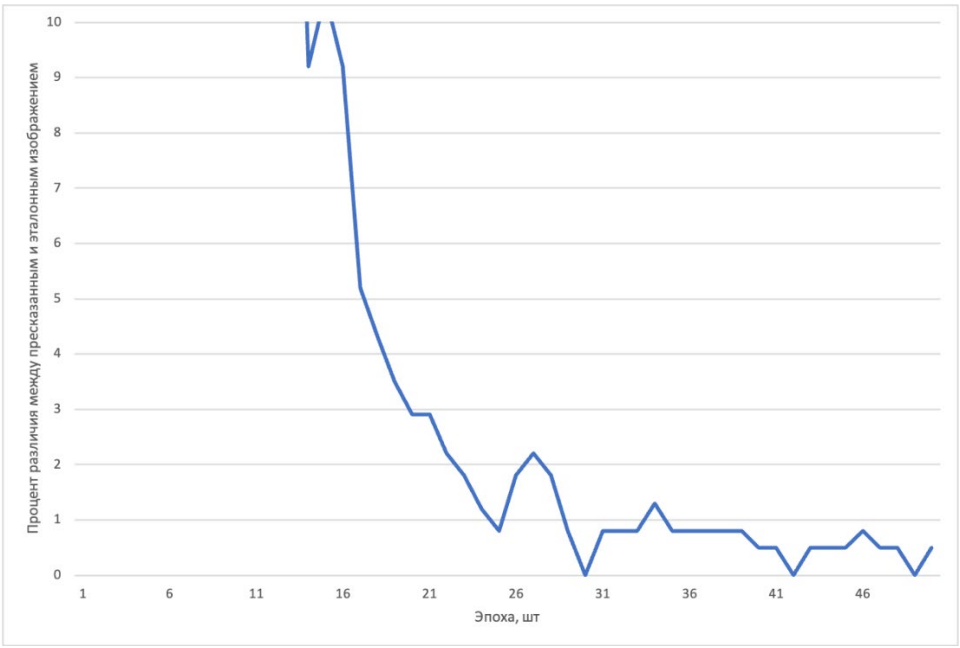

Рисунок 6 – Изменения результата выполнения функциональных тестов в процессе обучения с использованием архитектуры ResNet и Трансформеров
Figure 6 – Changes of functional test's result during training using ResNet and Transformer architecture

Результаты третьего эксперимента оказались лучше, чем у исходной и у двух ранее рассмотренных. Также, можно обратить внимание, что были эпохи, когда системе удалось точно предсказать необходимый код и воспроизвести то, что было на исходном изображении. Также стоит отметить, что в этот момент что значение метрик BLEU и chrF++, что процент различия изображений достигали своих крайних значений (1 для BLEU и chrF++ и 0 для процента различия изображений). Однако в рамках этого эксперимента можно увидеть тенденцию, которая появилась еще в рамках второго эксперимента: для обучения третьей системы и получения первого предсказания потребовалось 12 эпох, что в два раза больше, чем у исходной системы.

В результате проведенных экспериментов можно сделать вывод о том, что комбинация архитектур ResNet и Трансформеров позволяет достичь наиболее качественного результата при сравнении использования различных комбинаций этих архитектур. При этом, если рассматривать отдельно улучшение части, связанной с обработкой изображения, и части, связанной с обработкой текста, можно увидеть, что улучшения части по обработке текста показало результат лучше, чем обработка изображения. Также важно учитывать, что время, которое будет требоваться системе на обучение с использование архитектур ResNet и Трансформеров, становится больше.

Заключение

В ходе исследования была выдвинута гипотеза о том, что изменения архитектуры исходной системы позволит получить более высокие метрики качества. В процессе изучения материалов, было выдвинуто предложение использовать архитектуру ResNet для замены части, связанной с обработкой изображения, и архитектуры Трансформеров для замены части, связанной с обработкой текстов.

В рамках экспериментов было проведено сравнение показателей трех вариантов систем, модифицирующих систему, представленную в статье [6]. В рамках первого эксперимента обработка изображений заменена на архитектуру ResNet, в рамках второго эксперимента обработка текста с использованием LSTM заменена на архитектуру Трансформеров, в рамках третьего – используются одновременно ResNet и Трансформеры. В результате сравнения результатов трех экспериментов третий вариант показал себя наилучших образом, однако потребовал наибольшего количества времени для своего обучения.

В будущих исследованиях планируется продолжить заниматься улучшением системы, полученной в рамках третьего эксперимента. В частности, текущее обучение ограничено набором данных, представленным в рамках исследования оригинальной pix2code, что заключается в ограничении в максимальном предсказании в 150 символов. Используя все ранее полученные данные, представляется возможным проверить и оптимизировать работу системы таким образом, чтобы предсказывать более длинный исходный код.

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. Zhu Zh., Xue Zh., 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
3. Liu Ya., 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
4. 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

5. Никитин И.В. Влияние версии библиотеки TensorFlow на качество генерации кода по изображению. *Моделирование, оптимизация и информационные технологии.* 2024;12(4). https://doi.org/10.26102/2310-6018/2024.47.4.040
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.). https://doi.org/10.26102/2310-6018/2024.47.4.040
6. Никитин И.В. Оценка качества полученного результата в задаче генерации исходного кода по изображению. *Моделирование, оптимизация и информационные технологии*. 2025;13(1). https://doi.org/10.26102/2310-6018/2025.48.1.030
Nikitin I.V. Assessing the Quality of the Result in the Problem of Source Code Generation from an image. *Modeling, Optimization and Information Technology*. 2025;13(1). (In Russ.). https://doi.org/10.26102/2310-6018/2025.48.1.030
7. He K., Zhang X., Ren Sh., Sun J. Deep Residual Learning for Image Recognition. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 27–30 June 2016, Las Vegas, USA.* IEEE; 2016 pp. 770–778. https://doi.org/10.1109/CVPR.2016.90
8. Balduzzi D., Frean M., Leary L., Lewis J.P., Wan-Duo Ma K., McWilliams B. The Shattered Gradients Problem: If resnets are the answer, then what is the question? In: *ICML'17: Proceedings of the 34th International Conference on Machine Learning, 06–11 August 2017, Sydney, Australia*. 2017. pp. 342–350. https://doi.org/10.48550/arXiv.1702.08591
9. 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.
10. 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
11. Popović M. chrF++: Words Helping Character N-grams. In: *Proceedings of the Second Conference on Machine Translation, 07–08 September 2017, Copenhagen, Denmark.* Association for Computational Linguistics; 2017. pp. 612–618. https://doi.org/10.18653/v1/W17-4770

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, Postgraduate, Plekhanov Russian University of Economics, Moscow, the Russian Federation.

Статья поступила в редакцию 19.03.2025; одобрена после рецензирования 31.03.2025; принята к публикации 04.04.2025.

The article was submitted 19.03.2025; approved after reviewing 31.03.2025; accepted for publication 04.04.2025.