УДК 004.514
DOI: 10.26102/2310-6018/2025.51.4.017

# Разработка прототипа системы компьютерной автоматизированной диагностики с использованием платформы 3D Slicer

**А.С. Кружалов**✉

*Московский политехнический университет, Москва, Российская Федерация*

***Резюме.*** Статья посвящена разработке прототипа системы компьютерной автоматизированной диагностики для распознавания аневризм сосудов головного мозга с использованием платформы 3D Slicer. Актуальность работы обусловлена растущей нагрузкой на специалистов, занимающихся интерпретацией медицинских изображений, что требует автоматизации диагностических процессов для повышения качества оказания медицинской помощи. Важность предварительного прототипирования систем компьютерной автоматизированной диагностики на начальных этапах работы над системой определяется необходимостью проверки концепции системы и используемых алгоритмов, выявления потенциальных проблем и улучшения взаимодействия между техническими специалистами и экспертами в области медицины. В статье описываются ключевые аспекты разработки, включая использование открытых библиотек и плагинов, а также применение шаблонов проектирования для повышения гибкости и модульности программного кода. Основное внимание уделяется проектированию системы, включая архитектуру программного обеспечения, выбор используемых технологий и реализацию ключевых компонентов. Разработанный прототип системы позволяет пользователю выбирать изображения и модели распознавания, а также строить 3D-визуализации выделенных областей. Результаты работы демонстрируют эффективность предложенного подхода, а также возможности последующей интеграции разработанного прототипа с медицинскими информационными системами и системами архивации и передачи изображений (PACS).

***Ключевые слова:*** система компьютерной автоматизированной диагностики, прототипирование программного обеспечения, медицинская визуализация, 3D Slicer, искусственный интеллект в медицине.

***Для цитирования:*** Кружалов А.С. Разработка прототипа системы компьютерной автоматизированной диагностики с использованием платформы 3D Slicer. *Моделирование, оптимизация и информационные технологии.* 2025;13(4). URL: https://moitvivt.ru/ru/journal/pdf?id=2037 DOI: 10.26102/2310-6018/2025.51.4.017

# Development of a prototype of a computer-aided diagnostic system using 3D Slicer

**A.S. Kruzhalov**✉

*Moscow Polytechnic University, Moscow, the Russian Federation*

***Abstract.*** The article is devoted to the development of a prototype of a computer-aided diagnostics system for recognizing cerebral aneurysms using the 3D Slicer platform. The relevance of the work is due to the growing workload of specialists involved in the interpretation of medical images, which requires automation of diagnostic processes to improve the quality of medical care. The importance of prototyping computer-aided diagnostic systems at the initial stages of work on the system is determined by the need to test the concept of the system and the algorithms used, identify potential problems and improve interaction between technical specialists and experts in the field of medicine. The article describes key aspects of the development, including the use of open libraries and plugins, as well as the application of design patterns to increase the flexibility and modularity of the code. The main focus is

 

on the design of the system, including the software architecture, the choice of technologies used and the implementation of key components. The prototype of the system allows the user to select images and recognition models, as well as build 3D visualizations of the highlighted areas. The results of the work demonstrate the effectiveness of the proposed approach, as well as the possibilities of subsequent integration of the developed prototype with medical information systems and picture archiving and communication systems (PACS).

***Keywords:*** computer-aided diagnostics system, software prototyping, medical imaging, 3D Slicer, artificial intelligence in medicine.

***For citation:*** Kruzhalov A.S. Development of a prototype of a computer-aided diagnostic system using 3D Slicer. *Modeling, Optimization and Information Technology*. 2025;13(4). (In Russ.). URL: https://moitvivt.ru/ru/journal/pdf?id=2037 DOI: 10.26102/2310-6018/2025.51.4.017

## Введение

На сегодняшний день визуальный анализ медицинских изображений широко используется для диагностики заболеваний. Диагностическая визуализация играет ключевую роль в каждом медицинском учреждении и на всех уровнях здравоохранения [1]. Разработка систем компьютерной автоматизированной диагностики (КАД) в медицине крайне актуальна, так как такие системы помогают справиться с растущей нагрузкой на специалистов, анализирующих медицинские изображения, ускоряя обработку данных и снижая риск ошибок при высокой загруженности. Также системы КАД компенсируют нехватку квалифицированных кадров, особенно в удаленных регионах, предоставляя врачам мощные инструменты для диагностики даже при нехватке узкоспециализированного опыта. В результате это повышает доступность и качество медицинской помощи.

Также к преимуществам систем КАД можно отнести снижение субъективности в диагностике. Человеческий фактор, такой как усталость или недостаточный опыт, может привести к ошибкам, тогда как алгоритмы обеспечивают стабильность и воспроизводимость результатов. Кроме того, автоматизация рутинных задач позволяет врачам сосредоточиться на сложных случаях, требующих глубокого анализа и принятия решений.

Одной из основных задач, которую решают современные системы КАД, является выделение (сегментация) объекта интереса на изображении и его дальнейший морфометрический анализ. Задача сегментации в современных медицинских интеллектуальных системах, как правило, решается при помощи искусственных нейронных сетей (ИНС). Обучение ИНС – нетривиальный процесс, имеющий итерационную природу и требующий настройки большого количества параметров, поэтому при разработке системы распознавания медицинских изображений критически важна оценка результатов обучения ИНС как с точки зрения количественных показателей, так и с точки зрения качественного результата. Для этого необходима разработка специализированного программного обеспечения с графическим пользовательским интерфейсом, позволяющего построить визуализации полученных результатов. Разработка такого рода прототипа системы КАД на начальных этапах работы над системой существенным образом влияет на эффективность взаимодействия технических специалистов с экспертами в предметной области.

Цель данной работы – представить методику быстрого прототипирования системы компьютерной автоматизированной диагностики на базе платформы 3D Slicer с использованием модульного подхода, нейросетевых технологий и открытых библиотек, обеспечивающую переход от концепции к функциональному прототипу с возможностями сегментации, 3D-визуализации, расчета количественных характеристик.

## Материалы и методы

3D Slicer [2] – это бесплатная кроссплатформенная система с открытым исходным кодом для анализа, визуализации и обработки медицинских изображений. Разработка проекта была начата в 1998 г. как совместный проект между Surgical Planning Laboratory at the Brigham и Лабораторией искусственного интеллекта Массачусетского технологического института.

Среди основных достоинств 3D Slicer можно назвать следующие: расширяемость за счет возможности подключения плагинов, кроссплатформенность, возможность использовать в коммерческой разработке (распространяется по лицензии BSD), активное сообщество разработчиков, широкий набор встроенных возможностей, поддержка стандарта DICOM и современных систем архивации и передачи изображений (PACS), возможность расширения функциональности за счет написания собственных плагинов на языках программирования C++ и Python.

Ближайшим аналогом платформы 3D Slicer является проект The Medical Imaging Interaction Toolkit (MITK) [3]. MITK представляет собой систему с открытым исходным кодом для разработки интерактивных программ для обработки медицинских изображений. Как инструментарий, MITK предлагает те функции, которые не реализованы в таких библиотеках как ITK [4] или VTK [5]. MITK уступает 3D Slicer по набору встроенных функций и удобству использования.

3D Slicer – это исследовательское программное обеспечение, которое не проходило клинических испытаний, и поэтому не может быть использовано в клинической практике. С точки зрения прототипирования системы КАД это не является существенным ограничением, так как прототип призван продемонстрировать работоспособность системы на этапе разработки (первичная апробация разработанных алгоритмов), а итоговый программный продукт в любом случае должен проходить отдельные клинические испытания с целью дальнейшего внедрения.

Как видно из приведенного перечня основных преимуществ программы, она предоставляет исследователю широкие возможности по анализу и обработке медицинских изображений, причем многие из них доступны в виде подключаемых программных модулей (плагинов), которые можно использовать без написания программного кода.

Для разработки прототипа системы КАД наибольший интерес представляет возможность написания собственных плагинов на языках C++ и Python. Плагины в 3D Slicer – это пользовательские модули, которые расширяют функциональность платформы для решения специализированных задач (например, сегментации аневризм). Графический интерфейс плагинов реализуется при помощи библиотеки Qt[1]. При разработке плагинов могут быть использованы сторонние программные библиотеки (например, для работы с нейронными сетями).

В основе архитектуры разработанного прототипа лежат принципы модульности, гибкости и повторного использования кода. Эти принципы были реализованы через применение следующих основных шаблонов проектирования [6]: «Адаптер» для интеграции сторонних библиотек с 3D Slicer API, «Наблюдатель» для обработки событий GUI, «Стратегия» для гибкого выбора алгоритмов сегментации и «Фабрика» для создания экземпляров нейросетевых моделей сегментации. Перечисленные шаблоны реализованы через систему классов-прослоек (SegmenterLib), обеспечивающих модульность, инкапсуляцию логики обработки изображений и единые интерфейсы

[1] Qt Documentation. URL: https://doc.qt.io/ (дата обращения: 25.07.2025).

взаимодействия. Более подробное описание аспектов реализации упомянутых паттернов и детали архитектуры программного обеспечения приведены в следующем разделе.

## Результаты и обсуждение

В данной работе процесс прототипирования системы КАД с помощью платформы 3D Slicer рассматривается на примере разработки системы для распознавания аневризм сосудов головного мозга. Решается задача сегментации аневризм: на вход системе подается ангиограмма сосудов головного мозга, а на выходе пользователь получает маску сегментации, определяющую границы и местоположение объекта интереса (аневризмы). В качестве алгоритма распознавания используется ИНС на базе архитектуры U-Net [7]. Подробное описание используемой архитектуры нейронной сети и аспектов ее обучения, количественные оценки результатов распознавания приведены в работах [8, 9]. В качестве основной метрики для оценки качества результатов сегментации использовалась метрика Dice Score (DSC). В работе использовалась модель с DSC = 0,85. Эта оценка была получена на основе тестовой выборки. Тестовая выборка была сформирована путем случайного выбора изображений из имеющихся данных. В нее вошли 20 изображений, полученных с помощью ротационной ангиографии, на которых представлены примеры аневризм различных форм и размеров. При построении модели в качестве критерия достаточного качества сегментации было использовано условие достижения DSC > 0,8 на отложенной тестовой выборке.

Разработанное расширение предоставляет инструмент для автоматической сегментации церебральных аневризм на ангиограммах с использованием нейросетевой модели U-Net, включая 3D-визуализацию результатов, расчет морфометрических параметров и интерактивный интерфейс для работы с медицинскими изображениями (Рисунок 1).

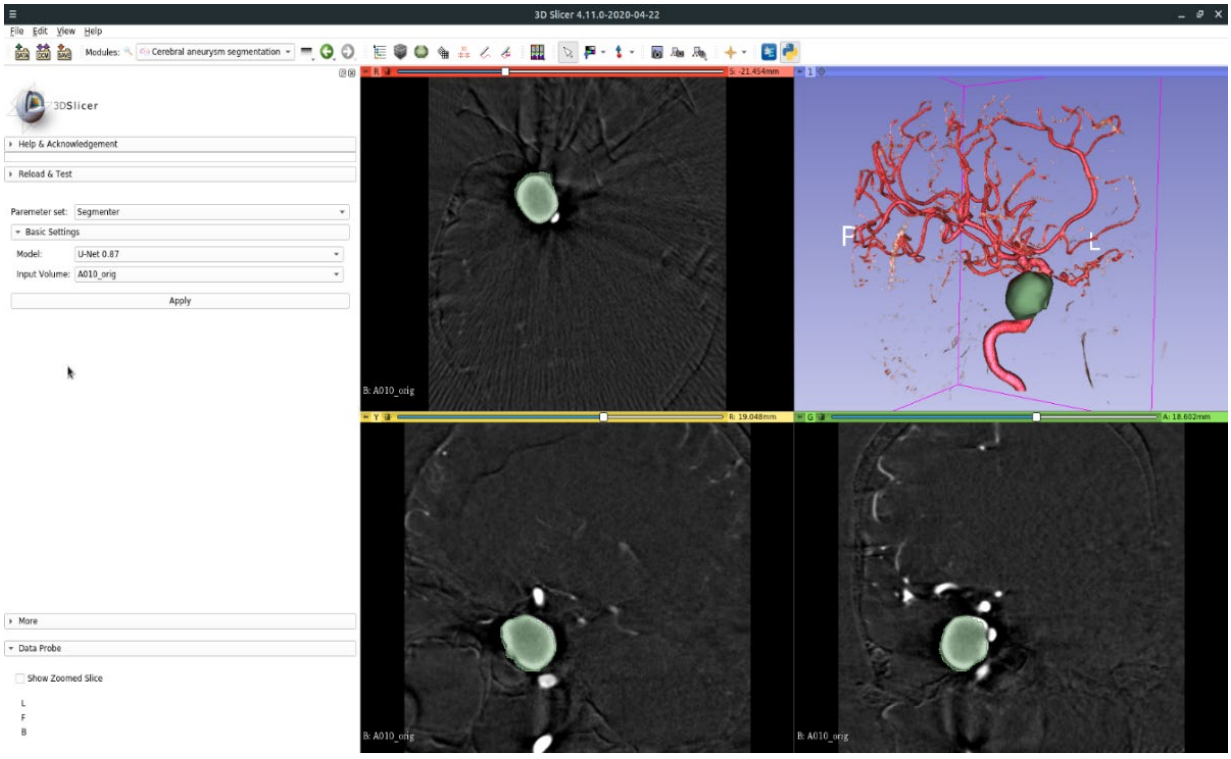


Рисунок 1 – Интерфейс расширения для программы 3D Slicer
Figure 1 – Extension interface for 3D Slicer

Расширение было написано на языке программирования Python версии 3.10 с использованием следующих программных библиотек: PyTorch [10], TorchIO [11], NumPy [12], scikit-image [13], SimpleITK [14], PyQt[1].

Структура директорий расширения должна соответствовать определенному шаблону (Рисунок 2).

В корневой директории должен находиться файл с метаданными для сборки расширения при помощи утилиты CMake, а также директория с основным кодом расширения (в приведенном примере – Segmenter), которая должна содержать директорию Resources для статичных файлов, директорию со вспомогательными программными модулями (SegmenterLib), а также основной файл с расширением .py (Segmenter.py) и файл CMake, в котором должен быть приведен перечень подключаемых при сборке файлов.

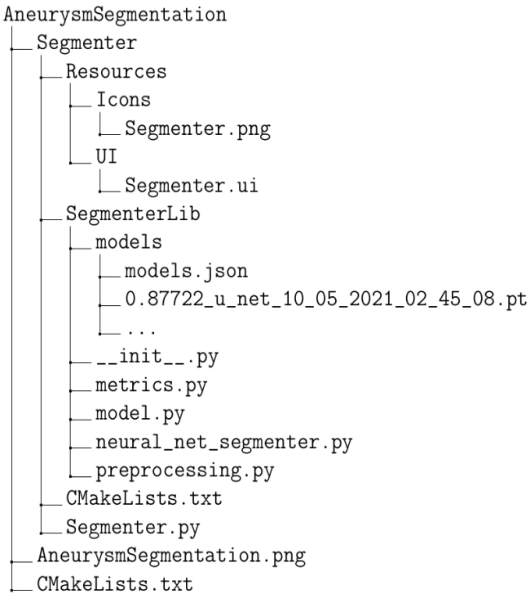


Рисунок 2 – Структура директорий и файлов расширения
Figure 2 – Extension directory and file structure

В директории Resources находятся статичные файлы с ресурсами расширения: значки и XML-файл, описывающий структуру GUI (graphical user interface) для используемого PyQt-виджета. Пользовательский интерфейс расширения состоит из набора различных полей ввода данных и элементов для отображения информации пользователю (Рисунок 3). Для проектирования интерфейса была использована программа Qt Designer.


Рисунок 3 – Внешний вид PyQt-виджета расширения
Figure 3 – PyQt extension widget appearance

Для создания расширения в 3D Slicer разработчик должен определить три ключевых класса: класс модуля (наследующий ScriptedLoadableModule) для регистрации расширения и указания метаданных, класс виджета (на базе ScriptedLoadableModuleWidget) для создания графического интерфейса и обработки действий пользователя, и класс логики (ScriptedLoadableModuleLogic), реализующий алгоритмы обработки данных (например, сегментацию U-Net). Наследование позволяет не реализовывать самостоятельно интеграцию расширения с другими программными модулями 3D Slicer, так как она уже реализована за счет интерфейсов, которыми обладают базовые классы, и разработчику остается только переопределить те методы, которые отвечают за функциональность разрабатываемого расширения. Также здесь используется шаблон проектирования «Адаптер» [14], который позволяет объектам с несовместимыми интерфейсами работать вместе. Упомянутые выше классы выступают в роли прослойки между 3D Slicer и сторонними библиотеками и инструментами (например, PyTorch), обеспечивая единый интерфейс для взаимодействия с различными компонентами системы.

Класс модуля (ScriptedLoadableModule) служит точкой входа расширения, регистрируя его в 3D Slicer и задавая метаданные (название, категорию, авторов), а также обеспечивая базовую инициализацию. Класс виджета (ScriptedLoadableModuleWidget) создает пользовательский интерфейс на основе Qt, загружая элементы из .ui-файла и связывая кнопки/поля ввода с методами-обработчиками событий. При взаимодействии пользователя (например, нажатии кнопки «Запуск») виджет передает данные (выбранное изображение, параметры) в класс логики (ScriptedLoadableModuleLogic), который выполняет основную обработку (сегментацию, анализ) с помощью внешних библиотек (PyTorch, ITK и др.). Виджет также обновляет интерфейс по завершении расчетов (например, отображает 3D-модель аневризмы или метрики точности распознавания). Такое разделение позволяет легко модифицировать интерфейс или алгоритмы независимо друг от друга, сохраняя код организованным и масштабируемым.

Набор методов класса «логики» строго не регламентирован и поэтому остается на усмотрение разработчика. Для разрабатываемой системы этот класс включает метод run, отвечающий за запуск сегментации, и набор вспомогательных методов для отображения результатов пользователю (например, метод для создания новой «сегментации» и построения на основе полученной маски 3D-визуализации выделенного объекта). Код, не связанный непосредственно с 3D Slicer, лучше размещать в отдельных модулях, чтобы его можно было впоследствии переиспользовать при необходимости. В частности, код, реализующий основную функциональность расширения, был размещен в отдельной директории SegmenterLib. Далее описан набор файлов, расположенных в данной директории.

В файле model.py приведена реализация класса нейронной сети на базе фреймворка PyTorch. Использование классов нейронных сетей с одинаковым интерфейсом делает их взаимозаменяемыми, обеспечивая возможность применения шаблона проектирования «Фабрика», который позволяет создавать объекты без указания конкретного класса создаваемого объекта, что упрощает процесс добавления новых моделей в систему.

В файле neural_net_segmenter.py расположен класс NeuralNetSegmenter, который инкапсулирует в себе преобразование формата изображения, инициализацию используемой модели, пред- и постобработку, вычисление итоговой маски сегментации. Инкапсулирование алгоритма сегментации в отдельный класс обеспечивает возможность использовать шаблон проектирования «Стратегия», который позволяет выбирать алгоритм сегментации во время выполнения программы. Это обеспечивает

гибкость и возможность легко добавлять новые алгоритмы без изменения существующего кода.

При запуске сегментации происходит следующая последовательность действий: данные изображений преобразуются в объекты изображений SITK, на основе которых создаются тензоры для последующей передачи данных для распознавания при помощи PyTorch-модели, модель и обрабатываемые данные переносятся в память GPU (в представленном прототипе инференс работает локально), происходит вычисление маски сегментации в виде PyTorch-тензора, который затем преобразуется в NumPy-массив, и на его основе производится расчет морфометрических характеристик (объем и др.) и создается интегрированный в 3D Slicer объект сегментации (vtkMRMLSegmentationNode), который поддерживает функцию построения 3D-визуализации (метод CreateClosed SurfaceRepresentation). В результате пользователю в интерфейсе платформы на трех проекциях отображается выделение области интереса и ее 3D-визуализация (Рисунок 1).

В файлах metrics.py и preprocessing.py реализованы вспомогательные функции: в metrics.py приведена реализация функций для вычисления используемых метрик точности распознавания (Dice Score), а в preprocessing.py приведены реализации функций для предварительной обработки изображений (пороговое преобразование, выделение наибольшей связной области и др.).

Для корректной работы разработанного программного обеспечения требуется компьютер с объемом оперативной памяти не менее 4 Гб, центральным процессором с частотой не менее 2 ГГц, графическим процессором с объемом видеопамяти не менее 4 Гб, поддерживающий технологию CUDA. Время обработки одного изображения зависит от размера изображения и технических характеристик используемой системы. На компьютере с процессором Intel Core i7-14700K с видеокартой NVIDIA GeForce RTX 3090 Ti время обработки изображения размером 220×256×256 вокселей составило 26 секунд.

## Заключение

Предложенный подход к разработке прототипа системы КАД демонстрирует эффективность модульной архитектуры, позволяя быстро создавать функциональные прототипы. Использование трех ключевых классов (модуля, виджета и логики) обеспечивает четкое разделение компонентов, упрощает интеграцию алгоритмов искусственного интеллекта и способствует дальнейшему внедрению решений в клиническую практику. Такой подход особенно ценен для исследовательских задач, где требуется гибкость и наглядная проверка концепций на ранних этапах.

Так как разработка прототипа рассматривалась с точки зрения первичной апробации разрабатываемых алгоритмов, не были рассмотрены некоторые аспекты, касающиеся дальнейшего внедрения системы в клиническую практику: необходимость реализации серверной обработки данных (так как для работы нейронной сети требуются существенные вычислительные ресурсы, как правило, недоступные на компьютере врача), интеграция системы с медицинскими информационными системами и системами архивации и передачи изображений (PACS) и др. Перечисленные функции могут рассматриваться как направления дальнейшего развития разработанного прототипа.

## СПИСОК ИСТОЧНИКОВ / REFERENCES


1. Осипов Л.В., Долгушин М.Б., Михайлов А.И. и др. Заглянуть в человека: визуализация в медицине. *Вестник Российского государственного медицинского университета.* 2016;(4):4–14.
   Osipov L.V., Dolgushin M.B., Mikhaylov A.I., et al. Looking Inside Man: Medical Imaging. *Bulletin of Russian State Medical University.* 2016;(4):4–13.
2. Kikinis R., Pieper S.D., Vosburgh K.G. 3D Slicer: A Platform for Subject-Specific Image Analysis, Visualization, and Clinical Support. In: *Intraoperative Imaging and Image-Guided Therapy.* New York: Springer; 2014. P. 277–289. https://doi.org/10.1007/978-1-4614-7657-3_19
3. Nolden M., Zelzer S., Seitel A., et al. The Medical Imaging Interaction Toolkit: Challenges and Advances. *International Journal of Computer Assisted Radiology and Surgery.* 2013;8(4):607–620. https://doi.org/10.1007/s11548-013-0840-8
4. McCormick M., Liu X., Jomier Ju., Marion Ch., Ibanez L. ITK: Enabling Reproducible Research and Open Science. *Frontiers in Neuroinformatics.* 2014;8. https://doi.org/10.3389/fninf.2014.00013
5. Schroeder W., Martin K., Lorensen B. *The Visualization Toolkit: An Object-Oriented Approach to 3D Graphics.* Kitware; 2006. 512 p.
6. Мартин Р. *Чистая архитектура. Искусство разработки программного обеспечения.* Санкт-Петербург: Питер; 2022. 352 с.
   Martin R.C. *Clean Architecture: A Craftsman's Guide to Software Structure and Design.* Saint Petersburg: Piter; 2022. 352 p. (In Russ.).
7. Ronneberger O., Fischer Ph., Brox Th. U-Net: Convolutional Networks for Biomedical Image Segmentation. In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015: Proceedings: Part III: 18th International Conference, 05–09 October 2015, Munich, Germany.* Cham: Springer; 2015. P. 234–241. https://doi.org/10.1007/978-3-319-24574-4_28
8. Кружалов А.С. Методика обучения свёрточной нейронной сети по фрагментам медицинских изображений в задаче распознавания церебральных аневризм. *Моделирование, оптимизация и информационные технологии.* 2023;11(2). https://doi.org/10.26102/2310-6018/2023.41.2.017
   Kruzhalov A.S. Patch-Based Training of a Convolutional Neural Network in the Problem of Cerebral Aneurysms Recognition. *Modeling, Optimization and Information Technology.* 2023;11(2). (In Russ.). https://doi.org/10.26102/2310-6018/2023.41.2.017
9. Кружалов А.С., Филиппович А.Ю. Особенности применения сверточных нейронных сетей для распознавания аневризм сосудов головного мозга. В сборнике: *Математические методы распознавания образов: Тезисы докладов 21-й Всероссийской конференции с международным участием, 12–15 декабря 2023 года, Москва, Россия.* Москва: Российская академия наук; 2023. С. 192–194.
10. Paszke A., Gross S., Massa F., et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems, 08–14 December 2019, Vancouver, BC, Canada.* New York: Curran Associates, Inc.; 2019. P. 8026–8037.
11. Pérez-García F., Sparks R., Ourselin S. TorchIO: A Python Library for Efficient Loading, Preprocessing, Augmentation and Patch-Based Sampling of Medical Images in Deep Learning. *Computer Methods and Programs in Biomedicine.* 2021;208. https://doi.org/10.1016/j.cmpb.2021.106236
12. Harris Ch.R., Millman K.J., van der Walt S.J., et al. Array Programming with NumPy. *Nature.* 2020;585(7825):357–362. https://doi.org/10.1038/s41586-020-2649-2

13. Van der Walt S., Schönberger J.L., Nunez-Iglesias Ju., et al. scikit-image: Image Processing in Python. *PeerJ.* 2014;2. https://doi.org/10.7717/peerj.453
14. Yaniv Z., Lowekamp B.C., Johnson H.J., Beare R. SimpleITK Image-Analysis Notebooks: A Collaborative Environment for Education and Reproducible Research. *Journal of Digital Imaging.* 2018;31(3):290–303. https://doi.org/10.1007/s10278-017-0037-8

## ИНФОРМАЦИЯ ОБ АВТОРАХ / INFORMATION ABOUT THE AUTHORS

**Кружалов Алексей Сергеевич,** старший преподаватель кафедры инфокогнитивных технологий, Московский политехнический университет, Москва, Российская Федерация.
*e-mail:* alexkruzhalov@gmail.com
ORCID: 0000-0003-0004-2334

**Alexey S. Kruzhalov,** Senior Lecturer at the Department of Infocognitive Technologies, Moscow Polytechnic University, Moscow, the Russian Federation.

*Статья поступила в редакцию 13.08.2025; одобрена после рецензирования 30.09.2025; принята к публикации 13.10.2025.*

*The article was submitted 13.08.2025; approved after reviewing 30.09.2025; accepted for publication 13.10.2025.*