УДК 519.68
DOI: 10.15827/0236-235X.137.014-019
Дата подачи статьи: 16.12.21
2022. Т. 35. № 1. С. 014–019

Архитектура программной платформы разработки и тестирования нейросетевых моделей для создания специализированных словарей

Д.Н. Пуртов [1], *аспирант, idmitry.purtov@gmail.com*
И.Г. Сидоркина [1], *д.т.н., профессор, декан, igs592000@mail.ru*

[1] *Поволжский государственный технологический университет, г. Йошкар-Ола, 424000, Россия*

Предложена реализация программной платформы для создания нейросетевых моделей с их тестированием, используемых для формирования специализированных словарей автоматизированных систем. Она позволяет ускорить процесс поиска оптимального метода для разработки нейросетевой модели. В основе платформы лежит обзор существующих инструментов и методов, используемых для создания моделей анализа текстов и технологий виртуализации ПО.

Авторами исследования разработана архитектура программной платформы для формирования специализированных словарей, обеспечивающая одновременное создание разных нейросетевых моделей в виртуальных контейнерах. Контейнерная виртуализация программных элементов, создающих и тестирующих нейросетевые модели, обеспечивает проведение всех математических расчетов по обработке текстовой информации, обучению и тестированию нейросетевой модели децентрализованно, параллельно и изолированно друг от друга. Обмен данными между виртуальными контейнерами, а также хранение результатов их работы осуществляются через специальную шину данных, представляющую собой дисковое пространство, к которому имеют доступ все контейнеры.

Применение разработанной платформы позволит ускорить процесс поиска алгоритма создания специализированных словарей через проверку гипотез, основанных на использовании различных методов построения моделей. Ускорение процесса происходит благодаря параллельности и повторному использованию математических результатов общих этапов алгоритмов, математические расчеты которых проведены похожим алгоритмом. Это позволяет масштабировать и дробить процесс обучения за счет параллельного создания различных моделей, а также на уровне отдельных этапов создания моделей. Предложенная платформа была успешно применена для поиска локально-оптимального метода создания модели в текстах узкой тематики.

Ключевые слова: *виртуализация среды, нейросетевая модель, python, sklearn, docker, модульная архитектура.*

Программные решения на базе нейросетевых моделей становятся все более востребованными. Доказано, что процесс разработки модулей с нейросетевой составляющей, например, для создания специализированных словарей, все более усложняется [1]. Это связано с неопределенностью [2] в работе моделей. Большую часть времени на их создание занимает выбор методов и параметров, по которым будут выполняться преобразование текстовой информации в числовую и обучение нейросетевой модели [2, 3].

Синтез оптимального решения по выбору параметров и методов построения нейросетевых моделей обусловливает необходимость экспериментировать с ними. Изменение параметров и методов приводит к изменению эффективности работы модели, количества математических операций и циклов обучения, что влияет на время, необходимое для создания модели [3, 4]. При реализации нейронной сети на языке программирования с использованием библиотеки sklearn [5] эти изменения заключаются в переписывании участка кода python, реализующего вызов методов из библиотеки.

Экспериментально доказано, что необходимость изменения параметров, а затем методов построения нейросетевых моделей при формировании специализированных словарей диктуется программной платформой, которая позволяет создавать множество различных моделей благодаря быстросменяемому и дополняемому программному коду при использовании специальной библиотеки, например, sklearn. В библиотеке для создания линейной нейросетевой модели на базе логистической регрессии достаточно вызвать метод sklearn.linear_model. LogisticRegression с нужными параметрами.

Результатом работы этой функции будет готовая модель для формирования специализированных словарей. Однако качество работы модели напрямую зависит от входных данных и параметров, переданных в метод. Например, параметр *C* в методе sklearn.linear_model. LogisticRegression настраивает силу регуляризации [5] и напрямую влияет на качество модели.

Библиотеки помогают быстро разрабатывать различные модели, ускоряя процесс подготовки данных и создания моделей с их оценкой. Ускорение достигается за счет того, что подобные библиотеки предоставляют готовые интерфейсы решения большинства задач, связанных с написанием программного кода. Однако остается проблема масштабирования процесса поиска локально-оптимального решения для формирования специализированных словарей. При этом доказано, что последовательный процесс подбора и комбинирования методов и параметров подготовки данных, а также создание и тестирование нейросетевой модели занимают достаточно много времени [6].

Архитектурно-инфраструктурные вопросы предлагается решать через программную платформу для поиска параметров и методов построения нейросетевых моделей, выполняющих узкоспециализированные задачи, например, создание модели для синтеза специализированных словарей, что позволит разрабатывать множество различных моделей параллельно, учитывая прошлый опыт и механизмы их построения, а также повторно использовать программный код.

Программная платформа для создания и тестирования нейросетевых моделей, решающих узкоспециализированные задачи, состоит из независимых блоков, работающих в изолированной виртуальной среде (рис. 1). Это позволяет создавать множество программных экземпляров, решающих одну и ту же задачу параллельно и осуществляющих обмен данными через специальную шину. Архитектура программной платформы в общем виде состоит из трех основных блоков: 1 – программный код, 2 – docker контейнеры, 3 – шина данных.

Блок с программным кодом представляет собой множество независимых узкоспециализированных программ или модулей, написанных на языке программирования. Наличие множества этих программ позволяет упростить их разработку и поддержку благодаря уменьшению функциональных особенностей программы [7]. Представленный блок менее подвержен рискам нестабильной работы и регрессионного эффекта изменения логики программы при внесении правок в программный код. Кроме этого, предложено группировать эти модули по их назначению, что позволит ускорить поиск нужных модулей. Данный блок с программным кодом для создания и тестирования нейросетевых моделей, формирующих специализированные словари, состоит из четырех групп модулей: обработки данных, создания моделей, оценки моделей, создания словарей.

Группа обработки данных включает модули, преобразующие текст в числовой вид [8], группа создания модулей – модули, обучающие нейросетевые модели. Модули оценки анализируют полученный результат [9], модули создания словарей формируют их по оценочным данным, исходя из того, является ли слово из текста искомым [9].

Блок docker контейнеров представляет собой виртуальные машины, которые могут содержать все необходимые предустановленные и настроенные программы [10]. В них сосредоточены модули, выполняющие программный код из первого блока.

Контейнерный подход позволяет запускать множество экземпляров как одних и тех же, так и разных модулей. Это решение [8, 11] позволяет распределять вычислительные мощности и параллельно работать с ними. Также контейнерный подход решает проблему настройки среды [10]. Это достигается за счет того, что контейнер является самодостаточной и независимой системой, включающей все необходимые дополнительные программные продукты и настройки. Таким образом, формируется кластер независимых виртуальных контейнеров с модулями, работающими параллельно в любой

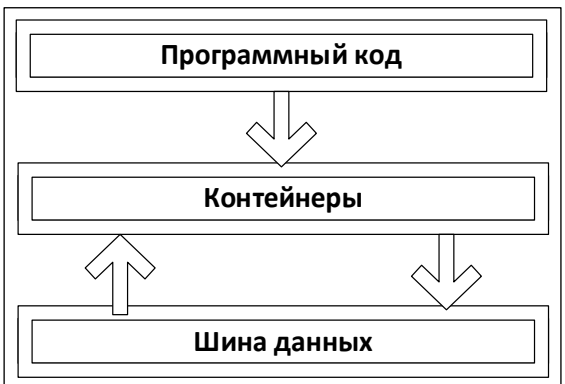

Рис. 1. Архитектура программной платформы

Fig. 1. A software platform architecture

локальной облачной среде, поддерживающей контейнеризацию.

Блок шины данных представляет собой общее дисковое пространство, которое можно получить, используя технологию монтирования папок [10] в docker контейнерах, что позволяет контейнерам обмениваться данными.

Таким образом, процесс создания нейросетевых моделей для формирования специализированных словарей состоит из следующих этапов: обработка данных, создание моделей, оценка моделей, создание словарей. Каждый этап выполняется в своем изолированном docker контейнере, формируя последовательность. Предложенная последовательность реализуется поэтапным запуском контейнеров с модулями и передачей между ними результатов работы на основе предложенных программных и архитектурных решений, как это показано на рисунке 2.

В результате последовательность создания и тестирования нейросетевых моделей с формированием специализированных словарей (рис. 2) реализует запуск модулей:

– обработки тестовых данных в контейнере с сохранением результата в шине данных (блок 1);

– создания нейросетевой модели с сохранением модели в шине данных (блок 2);

– оценки качества работы нейросетевой модели с сохранением результата оценки в шине данных (блок 3);

– создания узкоспециализированного словаря, в котором происходит обработка данных, полученных из других модулей, с принятием решения о формировании нужного словаря (блок 4).

Разработанная программная платформа позволяет создавать несколько последовательностей параллельно, что увеличивает скорость и дает возможность комбинировать существующие модели и их результаты для создания новой последовательности. Таким образом, число последовательностей для поиска локально-оптимального результата растет (рис. 3).

Предложенное решение параллельного создания последовательностей реализует методологию непрерывной интеграции [12] и раз-

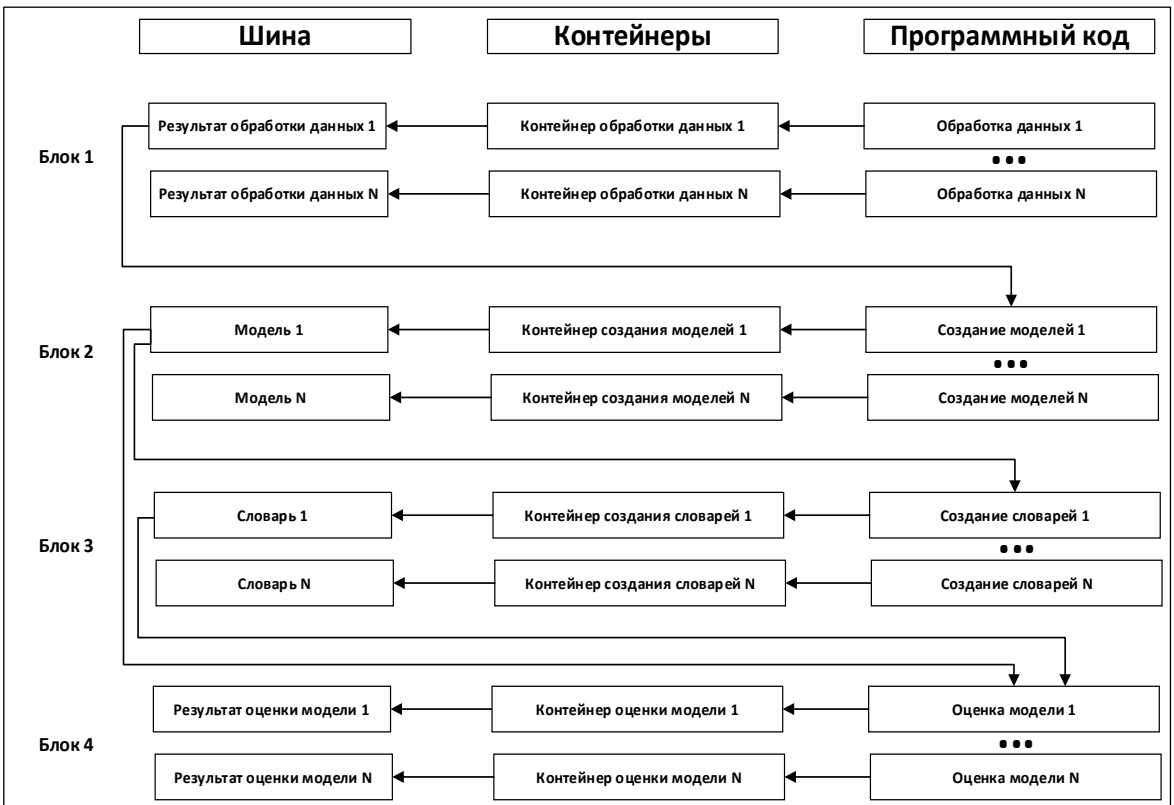

Рис. 2. Последовательность создания и тестирования нейросетевых моделей с формированием специализированных словарей

Fig. 2. A sequence of creating and testing neural network models with the formation of specialized dictionaries

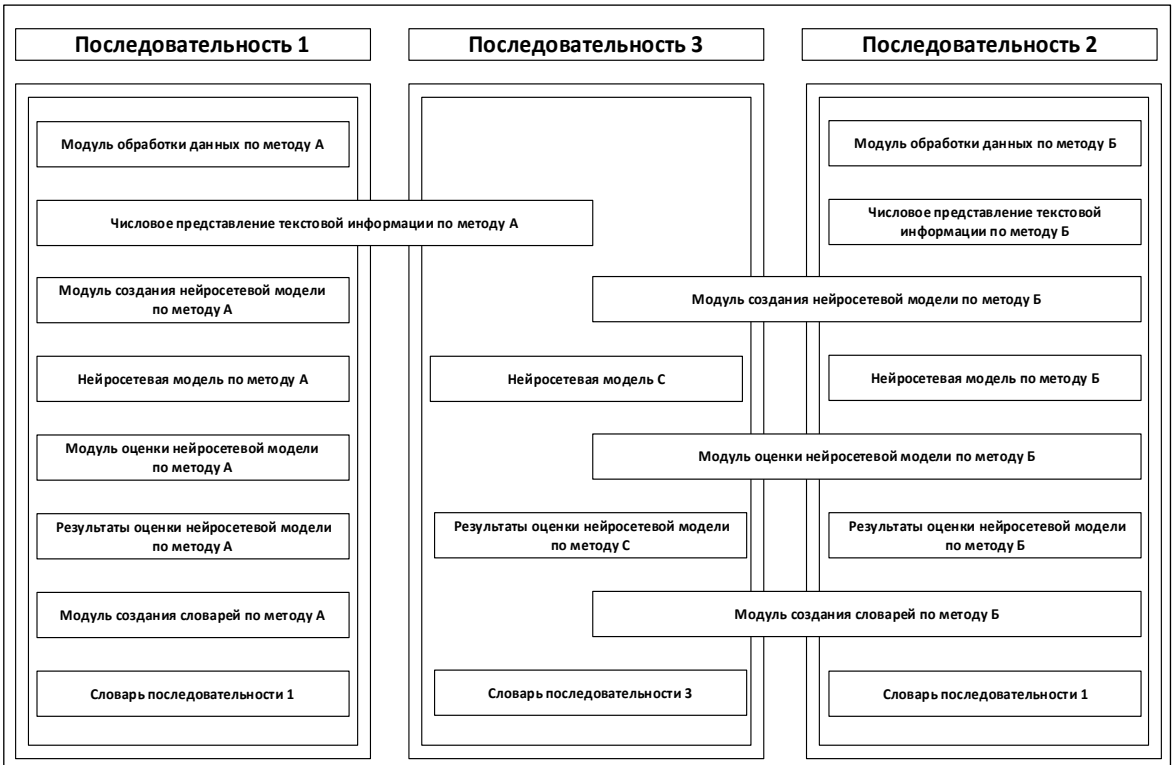

Рис. 3. Параллельное создание трех последовательностей разработки и тестирования нейросетевых моделей

Fig. 3. A parallel creation of three sequences of developing and testing neural network models

вертывания приложений с их выполнением. Параллельный запуск нескольких последовательностей позволяет создавать несколько решений одновременно для получения наилучшего локально-оптимального результата. Результатом каждого этапа последовательности является модуль для решения определенной задачи. Результат работы сохраняется в шине данных, что дает возможность повторно использовать его для создания новой последовательности через комбинирование с пропуском некоторых этапов. На рисунке 3 показано, как последовательность 3 создается из параллельно разработанных элементов последовательностей 1 и 2. Благодаря повторному использованию элементов уже существующих последовательностей и возможности строить любые из них параллельно уменьшается время создания новой последовательности.

Разработанная программная платформа решает архитектурно-инфраструктурные вопросы через программную платформу для поиска параметров и методов построения нейросетевых моделей, осуществляющих, например, создание модели для синтеза специализированных словарей. Платформа позволяет создавать множество различных моделей параллельно и повторно использовать программный код.

Таким образом, в результате исследования существующих инструментов и методов, используемых для создания моделей анализа текстов, предложены архитектура программной платформы и инструмент ее использования для формирования специализированных словарей. Важной особенностью программного решения является возможность одновременно создавать различные нейросетевые модели. Программная платформа позволяет строить последовательности разработки и тестирования нейросетевых моделей для формирования специализированных словарей, состоящие из следующих этапов: обработка данных, создание нейросетевой модели, оценка нейросетевой модели, создание словарей. Комбинирование этапов для ранее созданных последовательностей дает возможность получать новые для поиска локально-оптимального результата. Запуск всех последовательностей происходит в изолированной виртуальной среде параллельно и изолированно друг от друга, что позволяет увеличить скорость разработки разнообразных решений.

Литература

1. Фаустова К.И. Нейронные сети: Применение сегодня и перспективы развития // Территория науки. 2017. № 4. С. 83–87.
2. Гермиханова Х.Р. Методы обучения нейронной сети (некоторые аспекты) // Инновационные аспекты развития науки и техники: сб. статей. 2020. № 2. С. 6–10.
3. Гридин В.Н., Солодовников В.И., Карнаков В.В. Выбор начальных значений и оптимизация параметров нейронной сети // Новые информационные технологии в автоматизированных системах: матер. семинара. 2016. № 19. С. 270–273.
4. Пуртов Д.Н., Сидоркина И.Г. Проблема обучения нейронной сети при извлечении ключевой информации // ИС & ИТ. 2019. Ч. 2. С. 291–295.
5. Рашка С., Мирджалили В. Python и машинное обучение. Машинное и глубокое обучение с использованием Python, scikit-learn; [пер. с англ.]. СПб: Диалектика, 2020. 848 с.
6. Волосова А.В. Параллельные методы и алгоритмы. М.: Мади, 2020. 176 с.
7. Трембач В.М. Модульная архитектура интеллектуальной системы для решения задач интернета вещей // Открытое образование. 2019. № 3. С. 32–43. DOI: 10.21686/1818-4243-2019-4-32-43.
8. Жердева М.В., Артюшенко В.М. Стемминг и лемматизация в lucene.Net // Лесной вестник. 2016. № 3. С. 131–134.
9. Гуськов С.Ю., Лёвин В.В. Интервальные доверительные оценки для показателей качества бинарных классификаторов – ROC-кривых, AUC для случая малых выборок // Инженерный журнал: наука и инновации. 2015. № 3. С. 1–15. URL: http://engjournal.ru/catalog/mesc/idme/1376.html (дата обращения: 20.10.2021).
10. Mouat A. Using Docker: Developing and Deploying Software with Containers. O'Reilly Media Publ., 2017, 354 p.
11. Баранов А.В., Николаев Д.С. Использование контейнерной виртуализации в организации высокопроизводительных вычислений // Программные системы: теория и приложения. 2016. № 1. С. 117–134.
12. Шляпников В.М. Ускорение непрерывной интеграции и развертывания python-приложений // Инновационные аспекты развития науки и техники. 2021. № 2. С. 71–78.

Software & Systems Received 16.12.21
DOI: 10.15827/0236-235X.137.014-019 2022, vol. 35, no. 1, pp. 014–019

Architecture of the software development and testing platform neural network models for creating specialized dictionaries

D.N. Purtov [1], *Postgraduate Student, idmitry.purtov@gmail.com*
I.G. Sidorkina [1], *Dr.Sc. (Engineering), Professor, Dean, igs592000@mail.ru*

[1] *Volga State Technological University, Yoshkar-Ola, 424000, Russian Federation*

Abstract. The authors propose the implementation of a software platform for creating neural network models with their testing, used to create specialized dictionaries for automated systems. The software platform allows speeding up the process of finding the optimal method for creating a neural network model. The platform is based on an overview of existing tools and methods used to create clock analysis models and software virtualization technologies.

A research result is the proposed architecture of a software platform for creating specialized dictionaries that ensures the simultaneous creation of different neural network models in virtual containers. A container virtualization of software elements that create and test neural network models provides all mathematical calculations for processing text-based information; decentralized, in parallel and isolated training and testing a neural network model. The data exchange between virtual containers, as well as the storage of all the results of the container's operation occurs through a special data bus, which is disk space that all containers have access to.

The use of the developed platform can speed up the process of searching for an algorithm for creating specialized dictionaries through testing various hypotheses based on various methods for constructing models. The process acceleration occurs due to the parallelism and reuse of the mathematical results of the general

stages of algorithms whose mathematical calculations were carried out by a similar algorithm. This allows scaling and splitting the learning process not only through the parallel creation of various models, but also at the level of individual model creation stages. The proposed platform was successfully used to find a locally optimal method for creating a model in highly specialized limited-field texts.

Keywords: environment virtualization, neural network model, python, sklearn, docker, modular architecture.

References

1. Faustova K.I. Neural networks: Application today and development prospects. *Science Territory*, 2017, no. 4, pp. 83–87 (in Russ.).

2. Germikhanova Kh.R. Neural network training methods (some aspects). *Proc. Innovative Aspects of Science and Technology Development*, 2020, no. 2, pp. 6–10 (in Russ.).

3. Gridin V.N., Solodovnikov V.I., Karnakov V.V. Selection of initial values and optimization of neural network parameters. *Proc. New Information Technologies in Automated Systems*, 2016, no. 19, pp. 270–273 (in Russ.).

4. Purtov D.N., Sidorkina I.G. The problem of training a neural network when extracting key information. *IT & IS*, 2019, no. 2, pp. 291–295 (in Russ.).

5. Rashka S., Mirjalili V. *Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn*. 2020, 725 p. (Russ. ed.: St. Petersburg, 2020, 848 p.).

6. Volosova A.V. *Parallel Methods and Algorithms*. Moscow, 2020, 176 p. (in Russ.).

7. Trembach V.M. Modular architecture of an intelligent system for solving problems of the Internet of things. *Open Education*, 2019, no. 3, pp. 32–43. DOI: 10.21686/1818-4243-2019-4-32-43 (in Russ.).

8. Zherdeva M.V., Artyushenko V.M. Stemming and lemmatization in lucene.Net. *Forestry Bulletin*, 2016, no. 3, pp. 131–134 (in Russ.).

9. Guskov S.Yu., Levin V.V. Confidence interval estimation for quality factors of binary classifiers – roc curves, AUC for small samples. *Engineering Journal: Science and Innovations*, 2015, no. 3, pp. 1–15. Available at: http://engjournal.ru/catalog/mesc/idme/1376.html (accessed October 20, 2021) (in Russ.).

10. Mouat A. *Using Docker: Developing and Deploying Software with Containers*. O'Reilly Media Publ., 2017, 354 p.

11. Baranov A.V., Nikolaev D.S. The use of container virtualization in the organization of high-performance computing. *Program Systems: Theory and Applications*, 2016, no. 1, pp. 117–134 (in Russ.).

12. Shlyapnikov V.M. Accelerate continuous integration and deployment of python-applications. *Innovative Aspects of the Development of Science and Technology*, 2021, no. 2, pp. 71–78 (in Russ.).

Для цитирования

Пуртов Д.Н., Сидоркина И.Г. Архитектура программной платформы разработки и тестирования нейросетевых моделей для создания специализированных словарей // Программные продукты и системы. 2022. Т. 35. № 1. С. 014–019. DOI: 10.15827/0236-235X.137.014-019.

For citation

Purtov D.N., Sidorkina I.G. Architecture of the software development and testing platform neural network models for creating specialized dictionaries. *Software & Systems*, 2022, vol. 35, no. 1, pp. 014–019 (in Russ.). DOI: 10.15827/0236-235X.137.014-019.