ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»
(НИУ ВШЭ)

УТВЕРЖДАЮ

Руководитель
Исследовательского
центра в сфере
искусственного интеллекта
(должность)

Масютин А.А.
(ФИО)

«28» июня 2023 г.

М.П.

Программное обеспечение

«Библиотека программных и аналитических средств (фреймворка), направленных на предсказание расположения геномных функциональных элементов методами глубинного обучения на основе омиксных данных молекулярной биологии»

Описание программы

Лист утверждения

НИУ ВШЭ. 00003-01 13-ЛУ

СОГЛАСОВАНО

Заведующий
международной
лабораторией
биоинформатики
(должность)

Попцова М.С.
(ФИО)

«28» июня 2023 г.

Подпись и дата | Инв. № дубл. | Взам. инв. № | Подпись и дата | Инв. № подл.

Утвержден

НИУ ВШЭ.00003-01 13-ЛУ

Программное обеспечение

«Библиотека программных и аналитических средств (фреймворка), направленных на предсказание расположения геномных функциональных элементов методами глубинного обучения на основе омиксных данных молекулярной биологии»

Описание программы

НИУ ВШЭ.00003-01 13

Листов 20

2023 г.

Перв. примен.

Справ. N

Подп. и дата

Инв. N дубл.

Взам. инв. N

Подп. и дата

Инв. N подл.

Копировал

Аннотация

В данном документе приводятся общие сведения о программе, функциональное назначение, описание логической структуры, используемые технические средства, описание входных и выходных данных программного обеспечения «Библиотека программных и аналитических средств (фреймворка), направленных на предсказание расположения геномных функциональных элементов методами глубинного обучения на основе омиксных данных молекулярной биологии».

Оглавление

1. Общие сведения

1.1 Обозначение и наименование программы

«Библиотека программных и аналитических средств (фреймворка), направленных на предсказание расположения геномных функциональных элементов методами глубинного обучения на основе омиксных данных молекулярной биологии».

1.2 Программное обеспечение, необходимое для функционирования программы

Список дополнительного программного обеспечения и его конфигурации, необходимого для функционирования ПО, приведен в разделе 3.1 программного документа «Руководство системного программиста» на данную программу.

1.3 Языки программирования и дополнительное программное обеспечение

Программное обеспечение написано на языках программирования Python и R. Для инкапсуляции зависимостей используется система контейнеризации Docker; пользовательский интерфейс реализован с помощью JupyterLab.

Python – это высокоуровневый объектно-ориентированный язык программирования общего назначения. Основной функционал ПО для конфигурации и обучения моделей, а также для полногеномных предсказаний написан на Python.

JupyterLab – это интерактивная веб-среда разработки для блокнотов кода и данных. В рамках системы, JupyterLab обеспечивает доступ к шаблонам моделей для быстрого и простого прототипирования.

Docker – это система контейнеризации, которая позволяет упаковывать приложения и их зависимости в изолированные контейнеры, что обеспечивает их более надежную и устойчивую работу в различных средах. Docker используется для облегчения установки и эксплуатации системы.

2. Функциональное назначение

Программа предназначена для принятия решений в области предсказания геномных функциональных элементов с использованием нейросетевых моделей глубинного обучения разных типов архитектур в целях применения пользователем, не обладающим глубокими навыками программирования и/или не имеющего опыта применения алгоритмов искусственного интеллекта.

Программа реализует предсказание геномной разметки анализируемого ГФЭ на основе экспериментальных данных расположения этого ГФЭ в отдельных участках генома, а также других омиксных данных, подгружаемых пользователем с применением моделей на основе искусственного интеллекта.

В программе интегрированы пять функциональных модулей:

- Модуль №1 (Основной). «Предсказание ГФЭ на основе омиксных данных с использованием нейронных сетей разных типов, а именно сверточных, рекуррентных, гибридных сверточно-рекуррентных, генеративно-состязательных, графовых, а также нейронных сетей типа трансформер»
- Модуль №2. «Формирование и выборочный отбор омиксных данных с целью создания входных данных для моделей глубинного обучения»
- Модуль №3. «Анализ на ассоциацию геномных признаков с ГФЭ»
- Модуль №4. «Трансферное обучение для переноса аннотации ГФЭ»
- Модуль №5. «Интерпретация обученных нейросетевых моделей»

3. Описание логической структуры

3.1 Алгоритм программы

- Программное обеспечение начинает работу, когда пользователь запускает сервер JupyterLab и открывает пользовательскую среду в веб-браузере.
- Затем пользователь открывает шаблон, в котором, используя модуль №2, выбирает и загружает необходимые омиксные данные на сервер.
- Далее пользователь работает с Основным модулем, где ему предоставляется возможность выбора (1) омиксных данных необходимого типа из загруженных с помощью модуля №2, (2) аннотации ГФЭ для обучения, (3) полного генома анализируемого вида и (4) типа архитектуры нейронной сети для использования в предсказании.
- Пользователь запускает отредактированный им шаблон Основного модуля. Программа подгружает в память ранее загруженные омиксные данные и/или данные пользователя, конструирует модель и запускает процесс обучения модели. В процессе обучения Основной модуль визуализирует прогресс и текущие метрики.
- Программа использует обученную модель для полногеномного предсказания ГФЭ и предоставляет результаты предсказания пользователю.

3.2 Используемые принципы и методы разработки

При разработке использованы стандартные математические методы анализа данных, а также различные типы нейронных сетей, такие как сверточные, рекуррентные, гибридные сверточно-рекуррентные, графовые нейронные сети, а также нейронные сети типа трансформер. Основной архитектурной базой программы и ее компонентов является смесь объектно-ориентированного и функционального программирования.

1. Сверточные нейронные сети – алгоритм математической обработки изображений, представленных в машиночитаемом виде путем последовательного применения различных слоев, производящих трансформацию исходных данных изображения в целях получения итоговой метки класса, т.е. принадлежности изображения к той или иной категории, идентификации объектов на изображении и пр. Сверточные нейронные сети были успешно адаптированы для анализа геномных данных посредством их представления в виде числовых матриц, где значения полей матрицы кодируют последовательность ДНК и омиксные данные.
2. Рекуррентные нейронные сети – алгоритм математической обработки последовательностей данных любой длины. Рекуррентные сети учитывают контекст прошлых входных значений благодаря сохранению информации о предыдущих входах в виде скрытого состояния, которое передается на следующий шаг. Рекуррентные нейронные сети были успешно адаптированы для анализа геномных данных посредством их представления в виде последовательности нуклеотидов, а также сигналов расположения омиксных данных.
3. Гибридные сверточно-рекуррентные нейронные сети используют комбинацию сверточных и рекуррентных слоев, что позволяет

использовать преимущества обеих архитектур для улучшения качества классификации. Сверточные слои служат для извлечения признаков матрицы, составленной из оцифровки ДНК и омиксных данных, что позволяет выделить в матрице важные элементы и структуры, а рекуррентные слои применяются для учета контекста и последовательности данных, что позволяет учесть взаимосвязи между различными участками генома и использовать информацию о предыдущих состояниях сети для более точной классификации.

4. Генеративные нейронные сети – алгоритмы машинного обучения, задачей которых является восстановление исходного распределения данных и генерация новых данных из этого распределения. Они применяются для решения различных задач – например, аугментации данных, удаления шумов, обнаружения аномалий. Одной из наиболее перспективных архитектур генеративных нейронных сетей являются генеративно-состязательные сети. Генеративные модели успешно применяются в задачах геномики для генерации искусственных геномов, анализа функциональных частей генома и многих других задач.

5. Графовые нейронные сети – это модель машинного обучения, которая работает с графами – структурами данных, которые представляет объекты и связи между ними в виде узлов и ребер. Графовые нейронные сети используются для обработки данных, в которых объекты связаны между собой. Для графовых нейронных сетей геномная последовательность представляется в виде графа, а омиксные данные – в виде вектора свойств вершин графа. Такое представление позволяет по-разному обрабатывать омиксные данные, в зависимости от алгоритма нейронных сетей. Так во фреймворке реализованы графовые сверточные нейронные сети, графовые сети с механизмом внимания и графовые сети GraphSAGE.

6. Трансформер – специальная архитектура глубоких нейронных сетей глубинного обучения, предназначенная для обработки последовательностей. Отличительной особенностью архитектуры Трансформер является наличие механизма внимания. С его помощью можно находить взаимосвязи во входных данных на любом расстоянии (в отличие от сверточных и рекуррентных нейронных сетей), а также строить наглядные интерпретации получаемых результатов. В рамках данного фреймворка реализованы два вида архитектур типа трансформер. Один тип использует предобученную архитектуру DNABERT, которая работает только на геномных последовательностях без омиксных данных. Для включения информации об омиксных данных отдельно обучается нейросеть Трансформер, на вход которой наряду с представлением геномной последовательности подается вектор из омиксных признаков.

7. Трансферное обучение, в частности методы доменной адаптации могут быть использованы для переноса экспериментальных омиксных данных с одного генома на другой. В модуле доменной адаптации может быть использовано девять разных моделей: Domain Adversarial Neural Network (DANN), Deep Adaptation Network (DAN), Joint Adaptation Network (JAN), Adversarial Discriminative Domain Adaptation, Conditional Domain Adversarial Network (CDAN), Maximum Classifier Discrepancy (MCD), Adaptive Feature Norm (AFN), Margin Disparity Discrepancy (MDD), and Minimum Class Confusion (MCC).

8. Объектно-ориентированное программирование — это методология программирования, основанная на представлении программы в виде совокупности взаимодействующих объектов, каждый из которых является экземпляром определённого класса.

9. Функциональное программирование — это подход к разработке программного обеспечения, в котором функции рассматриваются как основные строительные блоки программы

3.3 Структура программы

Программа состоит из следующих элементов:

• JupyterLab – фронтенд, позволяющий пользователю производить загрузку данных на сервер, загрузку общедоступных омиксных данных, последовательностей полных геномов с удаленных серверов, а также выбор, конфигурацию и обучение моделей на пользовательских данных.

• Jupyter server – веб-сервер, напрямую взаимодействующий с файловой системой сервера и передающий команды на выполнение Python Kernel.

• Jupyter Kernel выполняет переданные команды, то есть исполняет Python, R или bash код, и возвращает результат в Jupyter Server, который, в свою очередь, предоставляет его пользователю через веб-браузер.

Файловая система сервера – заранее заданная структура директорий, в которых содержатся необходимые скрипты и объекты для запуска ПО.

3.3.1 Состав программы

Программа представлена как набор папок и файлов, формирующих единый дистрибутив:

- docker: папка, содержащая Dockerfile - файл с описанием программного окружения и зависимостей, необходимых для работы ПО
- DL_template: папка, обеспечивающая функционал основного Модуля №1. Включает папки:

- artifacts, содержащую результаты работы модуля – checkpoints моделей и полногеномные предсказания моделей;
- data, в которую пользователь должен загрузить исходные данные для построения моделей;
- assets, содержащую два поясняющих изображения для шаблона Template.ipynb;
- models, содержащую примеры различных типов нейронных сетей;

и файлы

- Template.ipynb, представляющий пример работы с модулем и осуществляющий запуск работы модуля через редактирование шаблона.
- README.mb с документацией модуля для пользователя

- OmicsDC: папка, обеспечивающая функционал Модуля №2. Включает папки:
  - OmicsDC, содержащую имплементацию функционала модуля загрузки данных
  - storage для сохранения результатов по умолчанию

  а также файлы:
  - README.mb с документацией модуля для пользователя
  - Example.ipynb, представляющий пример работы с модулем и осуществляющий запуск работы модуля через редактирование шаблона
- Association: папка, обеспечивающая функционал Модуля №3. Включает папки:
  - Association, содержащую имплементацию функционала модуля исследования признаков на ассоциацию
  - storage для сохранения результатов по умолчанию

- DomainAdaptation: папка, обеспечивающая функционал Модуля №4. Включает папки:
    - DomainAdaptation, содержащую имплементацию функционала модуля трансферного обучения
    - storage для сохранения результатов по умолчанию

  и файлы:

    - conda_requirements.yml с необходимыми библиотеками и их версиями для установки с помощью conda
    - pip.requirements с необходимыми библиотеками и их версиями для установки с помощью pip
- Interpretation: папка, обеспечивающая функционал Модуля №5. Включает файл:
    - lrp.py, содержащий имплементацию функционала модуля интерпретации обученных моделей на языке программирования Python

3.4 Соответствие программы стандартам

Программа соответствует стандарту PEP 8 (Python Enhancement Proposal - «Предложение по усовершенствованию Python»). PEP 8 представляет собой документ, содержащий рекомендации по написанию кода на Python.

4. Используемые технические средства

Для установки, запуска и стабильного функционирования программы требуется сервер со следующими минимальными системными требованиями:

- Количество физических ядер процессора (CPU) - 4;
- Оперативная память (RAM) - 32 Гб;
- Дисковое пространство (HDD) – 256 Гб.

5. Вызов и загрузка

5.1 Способ вызова

Для запуска программы требуется:

1. Зайти на сервер с помощью протокола SSH и установить SSH-соединение с переадресацией портов, используя следующую команду:

   ssh -L localhost:8888:localhost:8888 -p <server-port> <username>@<server-address>

   Перенаправить локальный порт 8888 на удаленный порт 8888. Заменить адрес сервера для подключения и номер порта.

2. Скачать репозиторий на сервер:

   git clone --recurse-submodules https://github.com/hse-bioinflab/framework

3. Создать контейнер Docker со всеми необходимыми зависимостями:

   cd docker

   docker build -t framework:1.0 .

   cd ..

4. Запустить интерфейс Jupyter Lab:

   *docker run -it --rm --runtime=nvidia --gpus all *

   *--shm-size=4GB -p 8888:8888 *

   *-v $(pwd):/workspace/ *

   framework:1.0

5. Проверить свою консоль на наличие URL-адреса, начинающегося с http://127.0.0.1:8888/lab?token =. Скопировать URL-адрес и открыть его в своем браузере.

6. Сохранить сеанс SSH запущенным во время работы.

5.2 Входные точки в программу

Входной точкой в программу является локальная ссылка http:/127.0.0.1:8080/.

6. Входные данные и выходные данные программы

Модуль №1. «Предсказание ГФЭ с использованием нейронных сетей разных типов, а именно сверточных, рекуррентных, гибридных сверточно-рекуррентных, генеративно-состязательных, графовых, а также нейронных сетей типа трансформер». Входные данные: загруженные пользователем файлы геномной аннотации в формате BED, загруженные геномные сборки FASTA и необходимые омиксные данные в формате BED. Выходные данные: файл в формате BigWig, где для каждой позиции генома выводится вероятность нахождения функционального элемента в данной позиции и файл с предсказаниями в формате BED.

Модуль №2. «Формирование и выборочный отбор омиксных данных с целью создания входных данных для моделей глубинного обучения». Входные данные: источник омиксных данных, набор параметров, определяющих критерии отбора омиксных данных». Выходные данные: архив, в котором содержатся файлы в формате BED с экспериментами, удовлетворяющие заданным параметрам.

Модуль №3. «Анализ на ассоциацию геномных признаков с ГФЭ». Входные данные: исследуемые геномные признаки в формате BED; файл chrom.sizes с размерами хромосом для исследуемого генома. Выходные данные: таблица с результатами статистического анализа.

Модуль №4. «Трансферное обучение для переноса аннотации ГФЭ». Входные данные: аннотация функционального элемента в исходном геноме в формате BED; полная последовательность генома, для которого требуется перенос аннотации. Выходные данные: аннотация участков в заданном геноме в формате BED.

Модуль №5. «Интерпретация обученных нейросетевых моделей». Входные данные: модель глубинного обучения, обученная для предсказания расположения геномных функциональных элементов; тензор, где для каждой позиции генома выводится вероятность нахождения функционального

элемента в данной позиции. Выходные данные: карта значимости омиксных данных, которая представляет из себя тензор уровня вклада каждого признака для каждой позиции генома: после усреднения по цепочке и по всей выборке можно получить усредненный уровень значимости в определении ГФЭ для каждого признака из омиксных данных.

7. Масштабирование и модернизация программы

Добавление новых методов в основной модуль ПО, таких как, например, обучение нейронных сетей других типов или моделей других архитектур, предполагается соответствующим редактированием шаблона основного модуля описанным ранее методом.

Лист регистрации изменений

Изм.	Номера листов (страниц)				Всего листов (страниц) в докум.	№ докум.	Входящий № сопроводительного докум. и дата	Подпись	Дата
	измененных	замененных	Новых	аннулированных					

Подп. и дата | Инв. N дубл. | Взам. инв. N | Подп. и дата | Инв. N подл.