Таким образом, задача классификации аудиозаписей биения сердца, с целью выявления патологий, является решаемой с помощью нейронных сетей. Тестирование созданной системы показало достаточно высокую прогностическую способность (75%) на примерах, не входящих в обучающую выборку. Стоит отметить, что данная система не рассматривается как самостоятельное решение диагностики, а служит для профилактического мониторинга пациентом собственных показателей.

1. Николенко С., Кадури А., Архангельская Е. Глубокое обучение. — СПб.: Питер, 2018. — 480 с.: ил. — (Серия «Библиотека программиста»).
2. Луис Педро Коэльо, Вилли Ричарт. Построение систем машинного обучения на языке Python. 2-е издание/ пер. с англ. Слинки А.А. - М.: ДМК Пресс, 2016. - 302 : ил
3. Плас Дж. Вандер. Python для сложных задач: наука о данных и машинное обучение. — СПб.: Питер, 2018. — 576 с.: ил. — (Серия «Бестселлеры O'Reilly»).
4. Сорокин А.Б. Искусственные нейронные сети прямого распространения: учебно-методическое пособие / Сорокин А.Б., О.В. Платонова. – М.: МИРЭА – Российский технологический университет, 2018. – 68с.

Гонатаев Р.Г., Омельченко Д.А., Фешина Е.В.

Преимущества разработки веб-приложений с применением фреймворков

*ФГБОУ ВО «Кубанский государственный аграрный университет имени И.Т. Трубилина»
(Россия, Краснодар)*

doi: 10.18411/lj-02-2021-03

idsp: ljjournal-02-2021-03

Аннотация

В статье проводится сравнение написания алгоритма на языке Python с другими языками программирования. Приводятся возможности использования фреймворков языка Python для создания сайтов.

Ключевые слова: фреймворк, язык программирования, Python, Django, Flask.

Abstract

This article compares writing an algorithm in Python with other programming languages. Possibilities of using the frameworks of the Python language for creating websites are given.

Keywords: framework, programming language, Python, Django, Flask.

Сейчас в индустрии разработки программного обеспечения существует очень большое количество инструментов, которые помогают разработчикам реализовывать желаемое. И, например, если вы хотите разрабатывать веб-приложения, писать под мобильные устройства, создавать десктопные приложения, то вы без труда можете найти языки программирования, которые предназначены для того или иного. Каждый инструмент для каждого разработчика удобен или не удобен по-своему, для этого существует много факторов, например, статическая или динамическая типизация у языка, как в языке реализуется инкапсуляция, сам синтаксис, какие библиотеки уже существуют.

Сегодня появилась тенденция универсальных языков, фреймворки которых позволяют расширить возможности языка почти под все что угодно. Если вы изучаете JavaScript, то это не значит, что вы все время будете писать обычные скрипты. Можно написать полноценный фронтенд, используя Vue.js или, допустим, React. Если, например, писать бэкенд, то для этого существует Node.js. Это и есть та самая универсальность языка, которая позволяет не заикливаться на одном направлении разработки и делать разнообразные приложения.

Одним из таких трендовых универсальных языков на сегодня является Python. Если разработчикам близка лаконичность языка, его читаемость, а также его ориентация на быстрое написание кода, то Python для разработок больше всего подходит. Создателем языка считается Гвидо ван Россум, голландский программист, который считается по праву создателем проекта. Работать над созданием данного языка он начал в далеких 80-х, когда для операционной системы Amoeba был необходим расширяемый скриптовый язык. Некоторые разработки Гвидо позаимствовал из ABC, а в феврале 1991 года уже был выложен исходный код в новостной группе. Одной из главных особенностей развития языка считается то, что большое сообщество помогает развивать данный язык, а также участвует в согласовании нововведений. У языка есть своя философия программирования, которая была создана Тимом Петерсом и называется «The Zen of Python». Вот некоторые отрывки:

- явное лучше, чем неявное;
- простое лучше, чем сложное;
- читаемость имеет значение;
- если реализацию сложно объяснить – идея плоха;
- если реализацию легко объяснить – идея, возможно, хороша.

Самое первое, что бросается в глаза при чтении кода, написанном на данном языке, это то, что вместо использования операторных скобок, используются пробелы, а также отсутствие точек с запятой. И в целом, синтаксические конструкции, которые позволяет реализовать Python, принципы разработки кода на данном языке делают написанный код максимально лаконичным и легко читаемым. В таблице 1 приведены примеры алгоритма, который делает одно и то же, только записан на разных языках программирования.

Если каждую команду алгоритма записывать отдельной строкой, то можно подсчитать, какое количество строк занимает код:

- Basic – 8 строк;
- Pascal – 11 строк;
- Алгоритмический язык – 11 строк;
- C++ – 11 строк;
- Python – 6 строк.

Таблица 1

Пример алгоритма, выполняющий одну и ту же задачу, но записанный на разных языках программирования

Язык программирования	Код
Basic	<pre> DIM S, N AS INTEGER S = 0 N = 170 WHILE S + N < 325 S = S + 25 N = N - 5 WEND PRINT S </pre>
Pascal	<pre> var s, n: integer; begin s := 0; n := 170; while s + n < 325 do begin s := s + 25; n := n - 5 end; writeln(s) end. </pre>

Алгоритмический язык	алг нач цел n, s s := 0 n := 170 нц пока s + n < 325 s := s + 25 n := n – 5 кц вывод s кон
C++	<pre>#include <iostream> using namespace std; int main() { int s = 0, n = 170; while (s + 2 < 325) { s = s + 25; n = n – 5; cout << s << endl; } return 0; }</pre>
Python	<pre>s = 0 n = 170 while s + 2 < 325: s += 25 n -= 5 print(s)</pre>

По приведенным выше данным можно определить, что Python занимает почти вдвое меньше строк кода алгоритма, чем C++ или Pascal, хотя этот код выполняет одно и то же. Даже человек, который не знаком с программированием, но хотя бы немного понимает английский язык и имеет навыки логики, сможет легко понять, что записано в тексте кода, написанном на Python. Именно в этом и кроется фантастическая популярность данного языка сегодня. Но необходимо понимать, что Python такой же инструмент, как C++. Если есть необходимость написать приложение, которое должно иметь высокую скорость работы и обработки данных, то Python в таком случае, не подойдет, потому что одна из его проблем как раз это его низкая скорость работы.

Фреймворки, созданные для данного языка, позволяют делать максимально разнообразные приложения, начиная от веб-технологий, анализа большого количества данных и заканчивая машинным обучением, например, такие как, Django, Flask, Tornado, Flask, FastApi, Pandas.

Django используется для разработки веб-приложения, как на языке шаблонов, так и для разработки REST API. Данный фреймворк использует паттерн MVC (Model – View – Controller), это означает, что логика приложения делится на три отдельных компонента. Основное преимущество данного паттерна заключается в том, что модификация любого из компонентов не затрагивает другие компоненты. Модель (Model) предоставляет данные и изменяется, реагируя на команды контроллера. Представление (View) отвечает за то, чтобы отображать данные пользователю и в ее функции входит реагировать на изменение модели. Контроллер (Controller) интерпретирует действия пользователя. При обращении к URL, контроллер обращается к определенному представлению, а оно, в свою очередь, вызывает определенную модель и отображает это пользователю. Один из основных принципов данного фреймворка – DRY (Don't repeat yourself).

Для работы с базой данных Django использует собственный ORM (Object Role Model). Данная технология позволяет обеспечивать работу с данными в терминах классов, по которым генерируется схема базы данных.

Данный веб-фреймворк используется в очень многих крупных и известных сайтах, как Instagram, YouTube, Google, Mozilla.

Возможности Django:

- встроенная ORM;
- встроенный интерфейс администратора;
- диспетчер URL, базирующийся на основе регулярных выражений;
- наличие системы шаблонов с наследованием и тегами;
- интернационализация;
- шаблоны функций контроллеров;
- наличие встроенной библиотеки для работы с формами;
- встроенная автоматическая документация.

Для того, чтобы создавать проект, построенный на API, был разработан Django Rest Framework (DRF) – библиотека, которая позволяет работать со стандартными моделями Django.

API DRF состоит из сериализатора, вида и маршрутизатора. Сериализатор (serializer) отвечает за преобразование информации, которая хранится в базе данных в JSON. Вид (ViewSet) определяет функции для создания CRUD интерфейса (create, read, update, delete). Маршрутизатор отвечает за URL-адреса, которые будут предоставлять доступ к ViewSets.

Также одним из популярных решения для разработки веб-приложений служит Flask. Это микрофреймворк для создания веб-приложений на языке Python. К нему можно подключить шаблонизатор Jinja2 для разработки приложений на шаблонах, а для разработки REST API понадобится Flask-Restful. Но если при создании проекта на Django, формируется большой каркас приложения, который необходимо дополнять и дописывать в зависимости от пользовательских потребностей, то во время разработки на Flask, необходимо писать все с «нуля», что может вызвать некоторые трудности. К тому же, во Flask, нет такого количества встроенных функций, как в Django, но это не является недостатком, наоборот, появляется возможность выбирать инструменты для разработки так, как это необходимо. Например, для работы с базами данных, необходимо подключать ORM. Самым частым решением является SQLAlchemy. Синтаксис построения запросов к базе очень похож на тот, который реализуется в Django, однако есть свои тонкости.

Также есть различия в работе между фреймворками при формировании миграций во время работы с базами данных. Если в Django существует встроенный функционал, который позволяет выполнить это, то во Flask, приходится подключать это вручную. Миграции позволяют сохранить структуру таблиц и переопределить ее в базе данных при необходимости, то есть отпадает нужда заново пересоздавать базу данных при каждом изменении ее структуры.

Несмотря на то, что существует большое количество различий между Flask и Django, нельзя сказать, что какой-либо из них является лучшим, а какой-то худшим. Большое приложение на Flask принимает структуру приложения на Django. Но инструменты, которые используются – разные. Каждый из этих фреймворков должен использоваться при необходимости и по потребностям.

1. Бэрри Пол. Изучаем программирование на Python /Бэрри Пол. – O'REILLY, 2017. – 611 с.
2. Мотылец А.А., Методы реализации веб-сайта в виде мобильного приложения./ А.А. Мотылец, Е.В. Фешина. Наука XXI века: проблемы, перспективы и актуальные во-просы развития общества: материалы международной межвузовской осенней научно-практической конференции, (пгт Яблоновский, 25 сентября 2020 года) – Издатель-ство: Краснодарский ЦНТИ – филиал ФГБУ «РЭА» Минэнерго России, 2020 – 333с.
3. Тузовский А. Ф. Проектирование и разработка web-приложений. Учебное пособие / А.Ф. Тузовский. – М.: Юрайт, 2016. – 220 с.
4. Фешина Е.В. Тенденции разработки мобильных приложений. / Д.А. Омельченко, В.Е. Меньшиков, Е.В. Фешина Тенденция разработки мобильных приложений. Информа-ционное общество: современное состояние и перспективы развития: сборник матери-алов XII международного студенческого форума. – Краснодар: КубГАУ, 2019. – 392с. С. 350-352.
5. Форсье, Дж. Django. Разработка веб-приложений на Python / Дж. Форсье. - М.: Сим-вол-плюс, 2014. - 343 с.