

МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ, ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ И ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ: ТЕОРИЯ И ПРАКТИКА

УДК 004.89

РАЗРАБОТКА СИСТЕМЫ РАСПОЗНАВАНИЯ ОБРАЗОВ НА PYTHON: АЛГОРИТМИЧЕСКИЙ ПОДХОД К АНАЛИЗУ БИНАРНЫХ ПРИЗНАКОВ

Бархатов Константин Сергеевич, студент
Арванова Саният Мухамедовна, старший преподаватель
Хатаев Абдурашид Мусаевич, студент
ФГБОУ ВО «КБГУ», г. Нальчик

DEVELOPMENT OF A PATTERN RECOGNITION SYSTEM IN PYTHON: AN ALGORITHMIC APPROACH TO BINARY FEATURE ANALYSIS

Barkhatov Konstantin Sergeevich, student, barhatov364@gmail.com
Arvanova Saniyat Mukhamedovna, Senior Lecturer, sani_07@mail.ru
Khatayev Abdurashid Musaevich, student, rasihatataev@mail.ru
Kabardino-Balkarian State University, Nalchik

Разработана система распознавания образов по бинарным признакам для оценки конфигураций. Используется многокритериальный анализ на основе семи метрик сходства. Решение ориентировано на задачи информационной безопасности

Ключевые слова: бинарные признаки, распознавание образов, метрики сходства, классификация, анализ данных, кибербезопасность.

A pattern recognition system based on binary features has been developed to evaluate system configurations. The approach applies multi-criteria analysis using seven similarity metrics. The solution is designed for applications in information security.

Keywords: binary features, pattern recognition, similarity metrics, classification, data analysis, cybersecurity.

В условиях цифровой трансформации и роста киберугроз важно разрабатывать методы анализа, учитывающие качественные признаки. Традиционные количественные подходы (размер файлов, частота запросов) не охватывают дискретные атрибуты, такие как функциональность ПО или поведенческие паттерны пользователей [1, с.112]. Для эффективного распознавания образов требуется унифицированный анализ разнородных данных с возможностью динамической классификации объектов при изменении их состояний.

Ключевая концепция базируется на пороговом принципе - объект изменяет состояние при достижении критического значения. Например, ОС становится «небезопасной» при нарушении конкретных условий (открытый исходный код → закрытый, многопользовательский режим → отключённый).

Методологическая основа – бинарное кодирование признаков. Любой объект X_k представляется вектором $X_k=(x_{k1}, x_{k2}, \dots, x_{kn})$, где $x_{kj}=1$ – наличие j -го признака, а $x_{kj}=0$ – его отсутствие.

Новизна исследования заключается в разработке адаптивной системы на Python для автоматизации преобразования объектов в бинарные векторы и реализации 7 алгоритмов сравнения для многокритериального анализа.

Разработанная программа представляет собой десктоп-приложение с графическим интерфейсом, реализованное на Python с использованием библиотеки Tkinter для визуального интерфейса. Система состоит из трёх ключевых модулей:

1. Модуль исходных данных – таблица бинарных признаков объектов.
2. Модуль попарного сравнения – расчёт 7 метрик сходства.
3. Аналитический модуль – агрегация результатов по методам.

На вкладке «Исходные данные» (Рис. 1) представлена матрица бинарных признаков для четырёх операционных систем. Каждая ОС описывается вектором X_k из 5 компонент, соответствующих критериям: многозадачность, многопользовательский режим, графический интерфейс, открытый исходный код, виртуализация. Это преобразование позволяет применять математический аппарат (метрики расстояний, алгоритмы кластеризации) к качественным данным.

Например, вектор для FreeBSD — [1,1,0,1,1], где отсутствие графического интерфейса ($x_{k3}=0$) отражает его минималистичный дизайн. Векторы других ОС формируются аналогично, позволяя их сравнивать.

Исходные данные						
Сравнение ОС						
ОС	Многозадачн	Многопольз	Графически	Открытый исх	Виртуализация	
Windows	1	1	1	0	1	
Linux	1	1	1	1	1	
macOS	1	1	1	0	1	
FreeBSD	1	1	0	1	1	

Рисунок 1. – Вкладка «Исходные данные»

На вкладке «Сравнение ОС» (Рис. 2) представлены результаты анализа схожести операционных систем по нескольким метрикам.

Исходные данные							
Сравнение ОС							
ОС1 vs ОС2	Рассел-Р	Жаккар-Н	Дайс	Сокаль-С	Сокаль-М	Кульжинс	Юл
Windows vs Linux	0.8000	0.8000	0.8889	0.6667	0.8000	0.4444	0.0000
Windows vs macOS	0.8000	1.0000	1.0000	1.0000	1.0000	0.5000	1.0000
Windows vs FreeBSD	0.6000	0.6000	0.7500	0.4286	0.6000	0.3750	-1.0000
Linux vs macOS	0.8000	0.8000	0.8889	0.6667	0.8000	0.4444	0.0000
Linux vs FreeBSD	0.8000	0.8000	0.8889	0.6667	0.8000	0.4444	0.0000
macOS vs FreeBSD	0.6000	0.6000	0.7500	0.4286	0.6000	0.3750	-1.0000

Рисунок 2. – Вкладка «Сравнение ОС»

В первой колонке указаны сравниваемые пары (Windows, Linux, macOS), а в остальных — коэффициенты схожести: Рассела-Рао, Жаккара-Ниджхёйса, Дайса, Сокаля-Снедека, Сокаля-Мишеля, Кульжинского и Юла. Значения от -1 до 1 отражают степень сходства, где более высокие указывают на большую схожесть, а отрицательные или нулевые — на значительные различия.

Итоговая вкладка (Рис. 3) содержит средние значения методов сравнения ОС. Метод Дайса имеет наивысший показатель (0.8611), отражая значительное сходство данных. Жаккара-Ниджхёйса и Сокаля-Мишнера дают одинаковый результат (0.7667), а наименьшее значение у Кульжинского (0.4306), что указывает на наибольшее различие ОС по этому методу.

Метод сравнения объектов реализуется через 7 функций, каждая из которых оперирует бинарными векторами, рассмотрим более детально.

Метрика Рассела-Рао оценивает долю совпадающих единиц в векторах. Этот метод подходит для оценки сходства, когда важна только доля совпадающих положительных признаков [2, с.9-10]. Обычно обозначают: a – число совпадений 1-1; n – общее число признаков (таблица).

Метрика Жаккара-Нидмена измеряет отношение числа совпадений 1-1 к количеству признаков, где хотя бы один из объектов имеет значение 1. Эта метрика подходит, когда важно учитывать только признаки, имеющие хотя бы одно положительное значения [3, с.125]. Обычно обозначают b – количество случаев, когда $x_i=1$ и $y_i=0$; c – количество случаев, когда $x_i=0$ и $y_i=1$ (таблица).

Метрика Дайса удваивает вес совпадающих единиц по сравнению с Жаккардом. Этот метод предпочтителен, когда критично учитывать совпадения 1-1 сильнее, чем различия (таблица).

Исходные данные	Сравнение ОС	Итоговые показатели
Метод сравнения		Средний показатель
Рассел-Рао		0.7333
Жокар-Нидмен		0.7667
Дайс		0.8611
Сокаль-Сниф		0.6429
Сокаль-Мишнер		0.7667
Кульжинский		0.4306
Юл		-0.1667

Рисунок 3. – Вкладка «Итоговые показатели»

Метрика Сокала-Снифа учитывает не только совпадения 1-1, но и различия. Она полезна для более сбалансированной оценки, учитывающей вклад различий. Метрика Сокала-Снифа представлена соотношением (таблица)

Программная реализация методов представлена таблицей.

Таблица – Программная реализация метрик

Название	Листинг программы	Формула
Метрика Рассела-Рао	<pre>def russell_rao(x, y): n = len(x) a = sum(1 for i in range(n) if x[i] == y[i] == 1) return a / n</pre>	$S_{rr} = \frac{a}{n} \quad (1)$
Метрика Жаккара-Нидмена	<pre>def jaccard_nidman(x, y): a = sum(1 for i in range(len(x)) if x[i] == y[i] == 1) b = sum(1 for i in range(len(x)) if x[i] == 1 and y[i] == 0) c = sum(1 for i in range(len(x)) if x[i] == 0 and y[i] == 1) return a / (a + b + c) if (a + b + c) != 0 else 0</pre>	$S_j = \frac{a}{a + b + c} \quad (2)$
Метрика Дайса	<pre>def dice(x, y): a = sum(1 for i in range(len(x)) if x[i] == y[i] == 1) b = sum(1 for i in range(len(x)) if x[i] == 1 and y[i] == 0) c = sum(1 for i in range(len(x)) if x[i] == 0 and y[i] == 1) return (2 * a) / (2 * a + b + c) if (2 * a + b + c) != 0 else 0</pre>	$S_d = \frac{2a}{2a + b + c} \quad (3)$
Метрика Сокала-Снифа	<pre>def sokal_sneath(x, y): a = sum(1 for i in range(len(x)) if x[i] == y[i] == 1) b = sum(1 for i in range(len(x)) if x[i] == 1 and y[i] == 0) c = sum(1 for i in range(len(x)) if x[i] == 0 and y[i] == 1) return a / (a + 2 * (b + c)) if (a + 2 * (b + c)) != 0 else 0</pre>	$S_{ss} = \frac{a}{a + 2(b + c)} \quad (4)$
Метрика Сокала-Мишнера	<pre>def sokal_michener(x, y): a = sum(1 for i in range(len(x)) if x[i] == y[i] == 1) d = sum(1 for i in range(len(x)) if x[i] == y[i] == 0) return (a + d) / len(x)</pre>	$S_{sm} = \frac{a + d}{a + b + c + d} \quad (5)$
Метрика Кульжинского	<pre>def kulczynski(x, y): a = sum(1 for i in range(len(x)) if x[i] == y[i] == 1) b = sum(1 for i in range(len(x)) if x[i] == 1 and y[i] == 0) c = sum(1 for i in range(len(x)) if x[i] == 0 and y[i] == 1) return a / (b + c) if (b + c) != 0 else 0</pre>	$S_k = \frac{a}{b + c} \quad (6)$
Метрика Юла	<pre>def yule(x, y): a = sum(1 for i in range(len(x)) if x[i] == y[i] == 1) b = sum(1 for i in range(len(x)) if x[i] == 1 and y[i] == 0) c = sum(1 for i in range(len(x)) if x[i] == 0 and y[i] == 1) d = sum(1 for i in range(len(x)) if x[i] == y[i] == 0) return (a * d - b * c) / (a * d + b * c) if (a * d + b * c) != 0 else 0</pre>	$S_y = \frac{ad - bc}{ad + bc} \quad (7)$

Метрика Сокала-Мишнера подходит для задач, где важно учитывать все типы совпадений, а также принимает во внимание и совпадения нулей, в формуле d – количество совпадений 0-0 (таблица 1).

Метрика Кульжинского оценивает степень преобладания совпадений 1-1 над различиями (таблица).

В свою очередь, метрика Юла интересна тем, что даёт высокие значения при сильной зависимости признаков. Юл использует корреляционное соотношение (таблица).

Каждая метрика отражает различные аспекты сходства. Методы Рассела-Рао, Жаккара и Дайса ориентированы на совпадения 1-1, тогда как Сокала-Мишнера учитывает и совпадения 0-0. Юл, в отличие от других, рассматривает статистическую взаимосвязь признаков.

Для кибербезопасности важно комбинировать несколько метрик, чтобы получить всесторонний анализ. В частности, метод Кульжинского позволяет обнаруживать значимые расхождения, а Юл – выявлять устойчивые закономерности [4, с.557].

Разработанная на Python система классифицирует объекты по бинарным признакам, что важно для анализа киберугроз и оценки конфигураций ПО. Использование семи метрик обеспечивает комплексный анализ данных. В будущем планируется интеграция машинного обучения для улучшения точности прогнозирования уязвимостей и адаптации к новым угрозам.

Список использованных источников

1. Ховард Р. Кибербезопасность: главные принципы. Обновленные стратегии и тактики. – СПб.: Питер, 2024. – 320 с.: ил. – (Серия «Библиотека программиста»). ISBN 978-5-4461-2201-1.
2. Александров Я.А., Сафин Л.К., Трошина К.Н., Чернов А.В. Статический бинарный анализ мобильных приложений для платформы Android по требованиям информационной безопасности. – 2023.
3. Шапиро С.Дж. Fancy Bear Goes Phishing: The Dark History of the Information Age, in Five Extraordinary Hacks. – Нью-Йорк: Farrar, Straus and Giroux, 2023. – 432 с. ISBN 978-0-374-60117-1.
4. Арванова, С. М., Ксенофонтов, А. С., Москаленко, Л. А. Криптографические механизмы безопасности // Научный альманах. – 2015. – № 7(9). – С. 578–580.