УДК 004.056.53

А.Г. Уймин [1]

[1] *РГУ нефти и газа (НИУ) имени И.М. Губкина, Российская Федерация*

ФУНКЦИОНАЛ ОБНАРУЖЕНИЯ ВНУТРЕННИХ УГРОЗ, ОСНОВАННЫЙ НА ДИНАМИКЕ МЫШИ

Аннотация. *Настоящая работа исследует проблему обнаружения внутренней угрозы через аутентификацию личности с использованием сети GoogLeNet и данных, сгенерированных на основе операций мыши. Мы предлагаем метод, основанный на сверточных нейронных сетях и анализе поведения мыши, для эффективного обнаружения внутренних угроз и поддельной аутентификации. Мы провели эксперименты на различных наборах данных с разным количеством базовых операций мыши. Результаты показали, что предложенный метод достигает высокой точности при обнаружении внутренних угроз. Наше исследование подтверждает потенциал использования сети GoogLeNet и анализа операций мыши в задаче обнаружения внутренней угрозы и предоставляет базу для дальнейших исследований в этой области.*

Ключевые слова: *Ключевые слова: обнаружение внутренней угрозы, аутентификация личности, сверточные нейронные сети, сеть GoogLeNet, анализ поведения мыши, базовые операции мыши, поддельная аутентификация, эксперимент, точность*

A. G. Uymin [1]

[1] *Russian State University of Oil and Gas (National Research University) named after I.M. Gubkin, Russian Federation*

DETECTION FUNCTIONALITY OF INTERNAL THREATS BASED ON MOUSE DYNAMICS

Abstract. *This study explores the problem of detecting insider threats through identity authentication using GoogLeNet network and mouse operation-generated data. We propose a method based on convolutional neural networks and mouse behavior analysis for effective detection of insider threats and fake authentication. We conducted experiments on various datasets with different numbers of basic mouse operations. The results showed that the proposed method achieves high accuracy in detecting insider threats. Our research confirms the potential of using GoogLeNet network and mouse operation analysis in the task of detecting insider threats and provides a basis for further research in this field.*

Keywords: *insider threat detection, identity authentication, convolutional neural networks, GoogLeNet network, mouse behavior analysis, basic mouse operations, fake authentication, experiment, accuracy.*

Введение

Обнаружение внутренних угроз в компьютерных системах является важной задачей для обеспечения безопасности информации в организации. Один из подходов к обнаружению внутренних угроз основан на анализе динамики мыши и применении методов глубокого обучения [1-8].

Этот подход основан на предположении, что стиль движения мыши пользователя может служить своеобразным "отпечатком" или "биометрическим" идентификатором, который можно использовать для распознавания и анализа действий пользователя [7]. Он предполагает сбор информации о движении мыши в режиме реального времени и последующий анализ этих данных с использованием методов глубокого обучения.

Процесс обнаружения внутренних угроз на основе динамики мыши и глубокого обучения обычно состоит из следующих шагов [8]:

Сбор данных: Собираются данные о движении мыши пользователя. Это может быть достигнуто путем регистрации координат и времени каждого события движения мыши, таких как перемещение, щелчки и скроллинг.

Предварительная обработка данных: Собранные данные подвергаются предварительной обработке для удаления шума и нормализации, чтобы обеспечить согласованность и сравнимость между разными пользователями.

Создание модели глубокого обучения: Используя предварительно обработанные данные, создается модель глубокого обучения, такая как рекуррентная нейронная сеть (RNN) или сверточная нейронная сеть (CNN). Модель обучается на собранных данных, чтобы изучить уникальные стили движения мыши каждого пользователя.

Обнаружение аномалий: После обучения модели глубокого обучения она может быть использована для обнаружения аномалий в динамике мыши. Если обнаруживается аномальное поведение, которое не соответствует стандартному стилю пользователя, система может считать это потенциальной внутренней угрозой и предпринять соответствующие меры, например, отправить предупреждение администратору системы или заблокировать доступ пользователя к конфиденциальным данным.

Предлагаемый метод, определяемый, "динамическим отображением мыши" [15], исследует основные операции, выполняемые мышью, такие как перемещение, щелчок, перетаскивание, прокрутка и остановка, с целью извлечения характеристик, отражающих поведение пользователей. Предыдущие исследования часто опирались на опыт исследователей при выборе и извлечении признаков из этих операций [10-14]. Например, расстояние перемещения, скорость перемещения и частота щелчков могут частично отражать поведение пользователей, использующих мышь. Однако существуют два основных недостатка в таком подходе. Во-первых, использование отдельной функции или их комбинации не всегда полностью или точно отображает уникальные поведенческие характеристики каждого человека. Во-вторых, при получении функций требуется больше времени, так как эффективные операции, основанные на базовых действиях, извлекаются из этих функций. Таким образом, требуется значительное количество базовых действий для создания достаточно эффективных операций.

Для полного сохранения функций, созданных пользователями при использовании мыши, предлагается применить метод сопоставления всех основных действий, выполненных пользователями, с изображениями. Поскольку данные, получаемые от мыши, представлены в виде одномерных наборов данных, мы используем метод отображения, который соотносит эти одномерные наборы данных с двумерными тензорами.

Общий алгоритм построения метода отображения

Шаг 1: Определение переменных:

m - количество базовых действий мыши.

xMax - максимальная координата x в сеансе.

yMax - максимальная координата y в сеансе.

Шаг 2: Построение системы координат D:

Создадим матрицу D размером (xMax+1) x (yMax+1), заполненную нулями, чтобы представить систему координат D.

Шаг 3: Извлечение операций "Перемещения":

Создадим пустую матрицу Move размером m x 2 для хранения координат перемещений.

Выполним m операций перемещения мыши и запишем каждую координату (x, y) в матрицу Move.

Отобразим красную цветную линию на матрице D, соответствующую координатам перемещений.

Шаг 4: Извлечение операций "Нажатия" и "Отпускания" кнопок мыши:

Создадим пустые матрицы LeftPress, LeftRelease, RightPress и RightRelease размером m x 2 для хранения координат операций нажатия и отпускания левой и правой кнопок мыши.

Запишем координаты операций нажатия левой кнопки в матрицу LeftPress, отпускания левой кнопки в матрицу LeftRelease, нажатия правой кнопки в матрицу RightPress и отпускания правой кнопки в матрицу RightRelease.

На матрице D отметим синим кругом нажатие левой кнопки, зеленым "x" отпускание левой кнопки, черным "." нажатие правой кнопки и черным "x" отпускание правой кнопки.

Шаг 5: Извлечение операций "Перетаскивания":

Создадим пустую матрицу Drag размером m x 2 для хранения координат операций перетаскивания.

Запишем координаты операций перетаскивания в матрицу Drag.

Отобразим толстую желтую линию на матрице D, представляющую операции перетаскивания.

Шаг 6: Извлечение операций "Прокрутки" (Scroll):

Создадим пустые матрицы ScrollUp и ScrollDown размером m x 2 для хранения координат операций прокрутки вверх и вниз соответственно. Запишем координаты операций прокрутки вверх в матрицу ScrollUp и координаты операций прокрутки вниз в матрицу ScrollDown.

На матрице D отметим голубым треугольником "△", направленным вверх, операции прокрутки вверх, и символом "▽" для операций прокрутки вниз.

Шаг 7: Извлечение операций "Пребывания":

Создадим пустую матрицу Stay размером m x 3 для хранения координат операций пребывания.

Запишем координаты операций пребывания и их относительное время в матрицу Stay.

На матрице D отобразим светло-красные полупрозрачные квадраты, размер которых линейно связан с относительным временем операции.

Шаг 8: Сохранение изображения D:

Сохраним матрицу D в виде изображения в формате JPG, чтобы получить диаграмму отслеживания поведения мыши в единицах базовых операций m.

Шаг 9: Повторение шагов 2-8:

Повторяем шаги 2-8, пока количество оставшихся операций в сеансе не станет меньше m.

Получаем n изображений пользовательского сеанса, представляющих последовательность действий мыши.

В соответствии с методом сопоставления, все сеансы пользователей сохраняются в базе данных в формализованных структурах Postgresql. Эти структуры данных представляют наборы операций мыши, выполненных каждым пользователем во время сеанса. Каждый сеанс идентифицируется уникальным идентификатором пользователя.

Для обучения моделей сверточных нейронных сетей (CNN) мы создали наборы изображений, используя данные из полученной базы двнных. Каждый набор изображений соответствует определенному значению m, которое представляет количество базовых действий мыши, равное 10, 20, 40, 80 и 800 соответственно.

Структура данных в Postgresql для хранения информации о сеансах пользователей и обработки предложенных шагов может быть организована с использованием двух таблиц: "Sessions" и "Actions". Пример описания структуры данных:

Таблица "Sessions":

• session_id (уникальный идентификатор сеанса)

• user_id (идентификатор пользователя)
• timestamp (временная метка сеанса)
Таблица "Actions":
• action_id (уникальный идентификатор операции)
• session_id (ссылка на сеанс, к которому относится операция)
• x_coordinate (координата x операции)
• y_coordinate (координата y операции)
• action_type (тип операции, например, "Move", "Click", "Drag", "Scroll", "Stay")
Пример SQL-кода для создания таблиц "Sessions" и "Actions" в базе данных Postgresql:

```
CREATE TABLE sessions (
  session_id SERIAL PRIMARY KEY,
  user_id INTEGER,
  timestamp TIMESTAMP
);

CREATE TABLE actions (
  action_id SERIAL PRIMARY KEY,
  session_id INTEGER REFERENCES sessions(session_id),
  x_coordinate INTEGER,
  y_coordinate INTEGER,
  action_type VARCHAR(50)
);
```

Для хранения данных в таблице "Actions" использованы операторы SQL INSERT, чтобы добавить записи для каждой операции мыши:

```
INSERT INTO actions (session_id, x_coordinate, y_coordinate, action_type)
VALUES (1, 100, 200, 'Move'),
     (1, 150, 250, 'Click'),
     (2, 200, 300, 'Drag');
```

Данные, полученные в ходе эксперимента сформированы в 100 наборов данных. Каждый набор данных был разделен на поднаборы с различным значением m (количество базовых действий мыши), такими как m=10, m=20, m=40, m=80 и m=800.

Каждый набор данных был помечен тегами FAR (%) и FRR (%), которые отражают процент ошибочных положительных результатов (False Acceptance Rate) и процент ошибочных отрицательных результатов (False Rejection Rate) соответственно.

Для увеличения набора данных был применен метод переворачивания изображений с вероятностью 50%. Это включало горизонтальное и вертикальное отражение каждого изображения. Каждое изображение было оценено с вероятностью 50% для определения, следует ли применять операцию переворачивания. Процесс увеличения данных продолжался до достижения заданной цели, после чего набор данных переставал увеличиваться.

Пусть:

m - количество базовых действий мыши,
N - общее количество наборов данных,
FAR(m) - False Acceptance Rate для значения m, формула 1
FRR(m) - False Rejection Rate для значения m, формула 2
FAR_avg - среднее значение FAR для всех наборов данных, формула 3
FRR_avg - среднее значение FRR для всех наборов данных формула 4.

Тогда формулы для вычисления FAR и FRR будут выглядеть следующим образом:

$$FAR(m) = (1/N) * \Sigma(FAR\_user\_i(m)) \quad (1)$$

, где i = 1 до N и FAR_user_i(m) - FAR для i-го пользователя в наборе данных с значением m.

FRR(m) = (1/N) * Σ(FRR_user_i(m)), (2)

где i = 1 до N и FRR_user_i(m) - FRR для i-го пользователя в наборе данных с значением m.

FAR_avg = (1/N) * Σ(FAR(m)), (3)

где m = 10, 20, 40, 80, 800.

FRR_avg = (1/N) * Σ(FRR(m)), (4)

где m = 10, 20, 40, 80, 800.

FAR_user_i(m) - False Acceptance Rate для i-го пользователя в наборе данных с значением m, может быть вычислен следующим образом:

FAR_user_i(m) = (количество ошибочных положительных результатов у пользователя i с m) / (количество всех попыток аутентификации у пользователя i с m) * 100, где "количество ошибочных положительных результатов у пользователя i с m" - количество случаев, когда система неправильно приняла неверные данные пользователя i при выполнении операций аутентификации с использованием m базовых действий мыши, а "количество всех попыток аутентификации у пользователя i с m" - общее количество попыток аутентификации, выполненных пользователем i с использованием m базовых действий мыши.

Аналогично, FRR_user_i(m) - False Rejection Rate для i-го пользователя в наборе данных с значением m, может быть вычислен следующим образом:

FRR_user_i(m) = (количество ошибочных отрицательных результатов у пользователя i с m) / (количество всех попыток аутентификации у пользователя i с m) * 100, где "количество ошибочных отрицательных результатов у пользователя i с m" - количество случаев, когда система неправильно отклонила правильные данные пользователя i при выполнении операций аутентификации с использованием m базовых действий мыши.

Таким образом, FAR(m) и FRR(m) для каждого значения m вычисляются как среднее значение FAR_user_i(m) и FRR_user_i(m) соответственно по всем пользователям в наборе данных.

А FAR_avg и FRR_avg - средние значения FAR и FRR для всех наборов данных, рассчитываются как среднее значение FAR(m) и FRR(m) соответственно по всем значениям m.

Сеть GoogLeNet, также известная как Inception, является глубокой нейронной сетью, разработанной исследователями компании Google в 2014 году. Она представляет собой одну из революционных архитектур для классификации изображений, основанную на сверточных нейронных сетях (CNN).

Основным преимуществом сети GoogLeNet является ее способность достичь высокой точности классификации при относительно небольшом количестве параметров. Она была разработана с использованием модульной структуры, в которой используются inception modules - блоки, объединяющие несколько сверточных слоев различных размеров фильтров, что позволяет сети эффективно извлекать информацию на разных уровнях детализации изображения.

Каждый модуль инцепции выполняет несколько параллельных операций свертки с разными фильтрами, а затем объединяет полученные результаты. Это позволяет сети GoogLeNet обрабатывать изображения на разных уровнях абстракции и извлекать более широкий спектр визуальных признаков.

В исследовании, сеть GoogLeNet использована для обработки сгенерированных изображений, представляющих поведение мыши в различных сеансах.

В рамках работы сети GoogLeNet мы построили 7-уровневую сверточную нейронную сеть (CNN), используя архитектуру GoogLeNet/Inception. Эта архитектура состоит из нескольких блоков, которые повторяются в сети, что позволяет сети извлекать различные уровни признаков и обрабатывать изображения на разных уровнях абстракции.

В каждом блоке сети GoogLeNet используется inception module, который объединяет несколько параллельных операций свертки с различными фильтрами и объединяет результаты. Это позволяет сети эффективно использовать информацию на разных уровнях детализации изображения. Каждый блок сети GoogLeNet включает сверточные слои, слои объедине-

ния (pooling layers), нормализацию, активацию и другие операции для обработки изображений. Сеть также содержит слои с глобальным усреднением (global average pooling) для уменьшения количества параметров и линейный слой для классификации.

Архитектура сети GoogLeNet позволяет ей достигать высокой точности классификации изображений при относительно небольшом количестве параметров. Благодаря параллельным операциям свертки и использованию inception module, сеть может эффективно обрабатывать изображения различной сложности и извлекать разнообразные визуальные признаки.

В рамках нашего эксперимента мы использовали следующие параметры:

Первый сверточный слой: 32 ядра размером 3x3 с шагом 1 пиксель. Он фильтрует входные изображения размером 100x100x3.

Второй сверточный слой: 64 ядра размером 3x3x2. Он принимает на вход выходные данные первого сверточного слоя.

Третий сверточный слой: 128 ядер размером 3x3.

В рамках нашего эксперимента мы использовали функцию максимального объединения (max pooling) в качестве функции объединения. Функция максимального объединения является распространенным методом объединения в сверточных нейронных сетях. Максимальное объединение производится путем выбора максимального значения в пределах заданной прямоугольной области (обычно размером zz) во входных данных. Это означает, что вместо вывода всего блока пикселей в этой области, мы выбираем только максимальное значение в этой области. Функция максимального объединения выполняется на каждом блоке пикселей с шагом S пикселей. Это позволяет снизить размерность данных и сделать представление более компактным, сохраняя важные признаки изображения. Таким образом, функция объединения помогает уменьшить количество параметров и вычислительную сложность сети, а также предотвращает переобучение.

В рамках функционала GoogLeNet, для предотвращения переобучения модели, был добавлен слой Dropout к первым двум полносвязным слоям. Dropout является методом регуляризации, который случайным образом исключает некоторые выходные данные узлов с определенной вероятностью во время обучения. Это помогает уменьшить сложную коадаптацию между нейронами и повысить обобщающую способность модели.

При использовании Dropout, каждый раунд обучения случайным образом исключает некоторые узлы, что эквивалентно созданию подсети или подмодели внутри сети. Удаленные узлы не участвуют в обратном распространении ошибки, что помогает предотвратить излишнюю зависимость между нейронами и снизить риск переобучения.

В нашем эксперименте, вероятность Dropout была установлена на 0,5, что означает, что в каждом раунде обучения случайным образом половина узлов исключается. Это позволяет эффективно регуляризовать модель, улучшить ее обобщающую способность и предотвратить переобучение.

В рамках функционала GoogLeNet, для градиентной оптимизации, мы использовали алгоритм Adam. Алгоритм Adam представляет собой оптимизационный метод с адаптивной скоростью обучения, который вводит коррекцию квадратичного градиента. Формулы 5-7

Обновление шага t:

t <- t + 1 (5)

Вычисление адаптивной скорости обучения lr_t:

lr_t <- learning_rate * sqrt((1 - бета_t) / (1 - бета_t^t)) (6)

Вычисление скользящего среднего первого момента m_t:

m_t <- beta1 * m_ + (1 - бета1) * g (7)

Вычисление скользящего среднего второго момента v_t:

v_t <- beta2 * v_ + (1 - бета2) * g^2 (7)

Обновление переменной с использованием адаптивной скорости обучения, скользящего среднего первого момента и скользящего среднего второго момента:

переменная <- переменная - lr_t * m_t / (sqrt(v_t) + epsilon)

В данном алгоритме:

learning_rate представляет скорость обучения (learning rate).

beta1 и beta2 являются коэффициентами сглаживания для скользящего среднего первого и второго моментов соответственно.

m_ и v_ представляют предыдущие значения скользящих средних первого и второго моментов.

g представляет градиент функции потерь по переменной.

epsilon - это малое число для численной стабильности, чтобы избежать деления на ноль.

Алгоритм Adam позволяет эффективно обновлять параметры модели с учетом скорости обучения, моментов градиента и адаптивной коррекции градиента. Это помогает ускорить сходимость модели и улучшить ее производительность в задаче оптимизации.

Мы предлагаем решение проблемы обнаружения внутренней угрозы через аутентификацию личности, где основной задачей является классификация пользователя как легитимного или нелегитимного. В рамках эксперимента, чтобы оценить эффективность нашего метода, мы выполняем следующие шаги.

Шаг 1. Набор данных изображений, сгенерированных ранее, используется. Определяется легитимный пользователь, и данные расширяются в соответствии с предыдущими шагами. Затем набор данных разделяется на обучающий набор T0 и тестовый набор T0' в соотношении 85% и 15%.

Шаг 2. Из T0 и T0' случайным образом извлекается одинаковое количество из подмножества других девяти пользователей для создания набора T1 для обучения нелегитимного пользователя и набора T1' для тестирования нелегитимных пользователей. Гарантируется отсутствие пересечения между T1 и T1'.

Шаг 3. Обучающий набор T0 и набор T1 используются в качестве входных данных для обучения заранее построенной модели сети CNN. Затем тестовый набор T0' и T1' проходят через обученную модель для оценки и вычисления FAR и FRR.

Шаг 4. Один из десяти пользователей назначается легитимным пользователем, а остальные девять считаются нелегитимными пользователями. Эксперимент повторяется снова, как описано выше, и вычисляются средние значения FAR и FRR.

В нашем эксперименте T0 + T0' = 5000 и T1 + T1' = 5000. Следовательно, размер обучающего набора равен T0 + T1 = 85%, а размер тестового набора равен T0 ' + T1' = 15%. Этот эксперимент проводится для разных наборов данных о работе мыши (m = 10, 20, 40, 80 или 800). Анализ показывает, что средние значения FAR и FRR зависят от значения m и уменьшаются с увеличением m (увеличение числа объектов на каждом изображении). Данные приведены в таблице 1

Таблица 1 – Данные FAR(%), FRR(%)

User Group	**m=10**		**m=20**		**m=40**		**m=80**		**m=800**	
	FAR(%)	FRR(%)	FAR(%)	FRR(%)	FAR(%)	FRR(%)	FAR(%)	FRR(%)	FAR(%)	FRR(%)
1	12.889	6.400	5.737	3.950	3.802	1.518	3.524	0.115	1.386	0.202
2	7.425	6.699	4.853	2.410	2.112	1.585	1.904	0.023	0.405	0.023
3	14.341	7.600	6.356	5.971	3.295	1.836	0.780	0.275	0.665	0.183
4	7.333	4.366	9.047	4.726	3.870	0.620	0.405	0.092	0.000	0.202
5	8.314	5.695	5.893	2.568	3.979	1.148	0.807	0.058	0.345	0.000
6	4.741	5.695	7.527	0.962	1.907	0.688	0.665	0.229	0.405	0.023
7	6.008	5.404	3.436	2.480	0.780	1.564	0.607	0.405	0.173	0.023
8	5.661	5.710	2.340	3.628	0.491	2.159	1.330	0.229	0.202	0.000
9	9.739	6.400	5.414	3.480	1.330	1.654	0.807	0.092	0.375	0.115
10	7.210	6.032	2.864	3.814	1.330	1.355	2.948	0.023	0.548	0.058
Avg	8.3672	6.1219	4.9436	3.5071	1.8317	1.4128	1.3975	0.1446	0.4509	0.0923

Средние значения метрик для каждого значения m составляют:

- FAR: 8.3672, 4.9436, 1.8317, 1.3975, 0.4509
- FRR: 6.1219, 3.5071, 1.4128, 0.1446, 0.0923

Из данных можно сделать следующие выводы:

1. С увеличением значения m уровень FAR и FRR снижается, что говорит о более точной и надежной аутентификации пользователей.
2. Для малых значений m (например, m=10) значения FAR и FRR являются более высокими, что может указывать на большую вероятность ошибок в аутентификации.
3. Среднее значение FAR и FRR уменьшается по мере увеличения значения m, что свидетельствует о повышении точности и надежности аутентификации при более длительном сборе и анализе данных о динамике мыши.
4. В целом, алгоритм аутентификации на основе данных о динамике мыши показывает низкий уровень ложных положительных и ложных отрицательных результатов, что делает его эффективным средством обеспечения безопасности и аутентификации пользователей.

В данном исследовании рассматривается пример усредненного пользователя, который генерирует приблизительно до 202 операций мыши в секунду. При использовании порога аутентификации m = 100, самое быстрое время для сбора данных и выполнения аутентификации составляет менее 1 секунды. Самый медленный случай характеризуется пользователем, выполняющим около 6 операций мыши в секунду, и для завершения сбора данных и аутентификации требуется примерно 1 минута.

В среднем, пользователи выполняют примерно 24,567 операций мыши в секунду, и время для завершения сбора данных и аутентификации составляет около 10 секунд при использовании порога аутентификации m = 100. В данном исследовании использовались исходные данные мыши.

Были проанализированы среднее время и минимальное время для различных значений порога аутентификации m (10, 20, 40, 80, 800). В таблице 2 представлены среднее и минимальное время для различных значений порога аутентификации m (10, 20, 40, 80, 800):

Таблица 2 - Среднее и минимальное время порога аутентификации m

m	**Среднее время (с)**	**Минимальное время (с)**
10	0.768	0.115
20	1.536	0.13
40	3.066	0.66
80	15.124	2.009
800	34.014	6.321

Эти значения указывают на среднее и минимальное время, необходимое для сбора данных и процесса аутентификации при заданных значениях порога m. Среднее время увеличивается с увеличением значения m, что указывает на выполнение и обработку большего количества операций. Аналогично, минимальное время также увеличивается, что говорит о возможности более продолжительного процесса аутентификации в некоторых случаях из-за сложности операций.

Заключение

Мы рассмотрели проблему обнаружения внутренней угрозы через аутентификацию личности с использованием сети GoogLeNet и набора данных, сгенерированного на основе операций мыши. Мы провели серию экспериментов, в которых оценивали эффективность нашего метода на разных наборах данных с различными значениями m (количество базовых операций мыши).

Исходя из результатов экспериментов, мы получили следующие выводы:

Наш метод аутентификации личности, основанный на сети GoogLeNet, показал общую эффективность в обнаружении внутренних угроз.

Средние значения FAR (частота ложных срабатываний) и FRR (частота ложных отрицаний) зависят от значения m (количество базовых операций мыши). С увеличением m, мы наблюдали снижение значений FAR и FRR.

Использование метода градиентной оптимизации Adam [4] позволило достичь хороших результатов обучения модели.

Применение функции максимального объединения в сверточных слоях сети GoogLeNet дало хорошую точность при классификации изображений операций мыши.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. 1. Antal M., Buza K., Fejer N. SapiAgent: A Bot Based on Deep Learning to Generate Human-Like Mouse Trajectories //IEEE Access. – 2021. – Т. 9. – С. 124396-124408.

2. Ciaramella G. et al. Continuous and Silent User Authentication Through Mouse Dynamics and Explainable Deep Learning //2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA). – IEEE, 2022. – С. 1791-1798.

3. Enström O. Authentication Using Deep Learning on User Generated Mouse Movement Images. – 2019.

4. Kingma D. P., Ba J. Adam: A method for stochastic optimization //arXiv preprint arXiv:1412.6980. – 2014.

5. McCann M. T., Jin K. H., Unser M. Convolutional neural networks for inverse problems in imaging: A review //IEEE Signal Processing Magazine. – 2017. – Т. 34. – №. 6. – С. 85-95.

6. N.A. Hamid, S. Safei, S.D.M. Satar, S. Chuprat, R. Ahmad (201 l). Mouse movement behavioral biometric systems. Proceedings of the International Conference on User Science and Engineering (i-USEr), Selangor, Malaysia, 29 November-l December 20ll, pp. 206-2ll.

7. Wei A., Zhao Y., Cai Z. A deep learning approach to web bot detection using mouse behavioral biometrics //Biometric Recognition: 14th Chinese Conference, CCBR 2019, Zhuzhou, China, October 12–13, 2019, Proceedings 14. – Springer International Publishing, 2019. – С. 388-395.

8. Y.X.M. Tan, A. Iacovazzi, I. Homoliak, Y. Elovici, A. Binder (20l9). Adversarial attacks on remote user authentication using behavioural mouse dynamics. Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, l4-l9 July 20l9, pp. l-l0.

9. Yuan S., Wu X. Deep learning for insider threat detection: Review, challenges and opportunities //Computers & Security. – 2021. – Т. 104. – С. 102221.

10.Zhou Y. T. et al. Image restoration using a neural network //IEEE transactions on acoustics, speech, and signal processing. – 1988. – Т. 36. – №. 7. – С. 1141-1151.

11.Есипов Д.А., Асланова Н., Шабала Е.Е., Щетинин Д.С., Попов И.Ю. МЕТОД ОБНАРУЖЕНИЯ ИНЦИДЕНТОВ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ ПО АНОМАЛИЯМ В БИОМЕТРИЧЕСКИХ ПОВЕДЕНЧЕСКИХ ЧЕРТАХ ПОЛЬЗОВАТЕЛЯ // Научно-технический вестник информационных технологий, механики и оптики. 2022. №4.

12.М. В. Тумбинская, Н. Ф. Асадуллин, Р. Р. Муртазин Моделирование аутентификации пользователей по динамике нажатий клавиш в промышленных автоматизированных системах // Программные продукты и системы. 2020. №2. URL:

13.Маҳкамов Анваржон Абдужабборович, Инадуллаев Холбек Ўрал Ўғли СРАВНИТЕЛЬНЫЙ АНАЛИЗ БИОМЕТРИЧЕСКИХ СИСТЕМ В ОБЕСПЕЧЕНИИ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ // Universum: технические науки. 2021. №12-1 (93).

14.Мельников Георг Андреевич, Карпук Анатолий Алексеевич ОБНАРУЖЕНИЕ ПОПЫТОК НЕСАНКЦИОНИРОВАННОГО ДОСТУПА НА ОСНОВЕ ВЫЯВЛЕНИЯ НЕСТАНДАРТНЫХ ПОВЕДЕНЧЕСКИХ ФАКТОРОВ // Universum: технические науки. 2021. №3-1 (84). URL:

15.Уймин А.Г., Морозов И.М. СРАВНИТЕЛЬНЫЙ АНАЛИЗ ИНСТРУМЕНТОВ НЕПРЕРЫВНОЙ ОНЛАЙН-АУТЕНТИФИКАЦИИ И СИСТЕМ ОБНАРУЖЕНИЯ АНОМАЛИЙ ДЛЯ ПОСТОЯННОГО ПОДТВЕРЖДЕНИЯ ЛИЧНОСТИ ПОЛЬЗОВАТЕЛЯ // T-Comm. 2022. №5.

REFERENCES

1. Antal M., Buza K., Fejer N. SapiAgent: A Bot Based on Deep Learning to Generate Human-Like Mouse Trajectories //IEEE Access. – 2021. – T. 9. – C. 124396-124408.

2. Ciaramella G. et al. Continuous and Silent User Authentication Through Mouse Dynamics and Explainable Deep Learning //2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA). – IEEE, 2022. – C. 1791-1798.

3. Enström O. Authentication Using Deep Learning on User Generated Mouse Movement Images. – 2019.

4. Kingma D. P., Ba J. Adam: A method for stochastic optimization //arXiv preprint arXiv:1412.6980. – 2014.

5. McCann M. T., Jin K. H., Unser M. Convolutional neural networks for inverse problems in imaging: A review //IEEE Signal Processing Magazine. – 2017. – T. 34. – №. 6. – C. 85-95.

6. N.A. Hamid, S. Safei, S.D.M. Satar, S. Chuprat, R. Ahmad (201 l). Mouse movement behavioral biometric systems. Proceedings of the International Conference on User Science and Engineering (i-USEr), Selangor, Malaysia, 29 November-l December 20ll, pp. 206-2ll.

7. Wei A., Zhao Y., Cai Z. A deep learning approach to web bot detection using mouse behavioral biometrics //Biometric Recognition: 14th Chinese Conference, CCBR 2019, Zhuzhou, China, October 12–13, 2019, Proceedings 14. – Springer International Publishing, 2019. – C. 388-395.

8. Y.X.M. Tan, A. Iacovazzi, I. Homoliak, Y. Elovici, A. Binder (20l9). Adversarial attacks on remote user authentication using behavioural mouse dynamics. Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, l4-l9 July 20l9, pp. l-l0.

9. Yuan S., Wu X. Deep learning for insider threat detection: Review, challenges and opportunities //Computers & Security. – 2021. – T. 104. – C. 102221.

10. Zhou Y. T. et al. Image restoration using a neural network //IEEE transactions on acoustics, speech, and signal processing. – 1988. – T. 36. – №. 7. – C. 1141-1151.

11. Esipov D.A., Aslanova N., Shabala E.E., Shchetinin D.S., Popov I.Yu. METHOD OF DETECTING INCIDENTS OF INFORMATION SECURITY BASED ON ANOMALIES IN USER BIOMETRIC BEHAVIOR CHARACTERISTICS // Scientific and Technical Bulletin of Information Technologies, Mechanics and Optics. 2022. No. 4.

12.Tumbinskaya M.V., Asadullin N.F., Murtazin R.R. MODELING USER AUTHENTICATION BASED ON KEYSTROKE DYNAMICS IN INDUSTRIAL AUTOMATED SYSTEMS // Software and Systems. 2020. No. 2. URL:

13.Makhkamov Anvarjon Abdudzhabborovich, Inadullaev Holbek Ural Ogly COMPARATIVE ANALYSIS OF BIOMETRIC SYSTEMS IN INFORMATION SECURITY PROVISION // Universum: Technical Sciences. 2021. No. 12-1 (93).

14.Melnikov Georg Andreevich, Karpuk Anatoly Alekseevich DETECTION OF UNAUTHORIZED ACCESS ATTEMPTS BASED ON DETECTING ABNORMAL BEHAVIORAL FACTORS // Universum: Technical Sciences. 2021. No. 3-1 (84). URL:

15.Uymin A.G., Morozov I.M. COMPARATIVE ANALYSIS OF TOOLS FOR CONTINUOUS ONLINE AUTHENTICATION AND ANOMALY DETECTION SYSTEMS FOR PERMANENT USER IDENTITY CONFIRMATION // T-Comm. 2022. No. 5.

Информация об авторах

Уймин Антон Григорьевич – аспирант профессор Владимирский Государственный Университет имени Александра Григорьевича и Николая Григорьевича Столетовых (ВлГУ),

ст. преподаватель кафедры безопасности информационных технологий РГУ нефти и газа (НИУ) имени И.М. Губкина, г. Москва, e-mail^ au-mail@ya.ru

Information about the authors

Anton Grigorievich Uymin is a graduate student and professor at Vladimir State University named after Alexander Grigorievich and Nikolai Grigorievich Stoletov (VlSU), senior lecturer at the Department of Information Security at Gubkin Russian State University of Oil and Gas (National Research University), Moscow, email: au-mail@ya.ru.