

УДК 004.93'12

А. В. Березникер¹, М. А. Казачук², И. В. Машечкин³, М. И. Петровский⁴,
И. С. Попов⁵

ДИНАМИЧЕСКАЯ АУТЕНТИФИКАЦИЯ ПОЛЬЗОВАТЕЛЕЙ НА ОСНОВЕ АНАЛИЗА РАБОТЫ С КОМПЬЮТЕРНОЙ МЫШЬЮ

Целью данной работы является исследование существующих и разработка собственных алгоритмов динамической аутентификации пользователей на основе анализа работы с компьютерной мышью, показывающих высокое качество работы и способных работать в динамическом режиме. В работе рассматриваются существующие способы построения и предобработки признакового пространства, а также методы динамической аутентификации, основанные на использовании классических методов машинного обучения и нейронных сетях. Для наиболее перспективных методов предлагаются модификации, обладающие более высокой эффективностью. На основе предложенного метода, демонстрирующего наилучшее качество работы, было разработано и реализовано кроссплатформенное приложение для динамической аутентификации пользователей на основе анализа работы с компьютерной мышью. Данная система и ее отдельные модули могут послужить основой для построения перспективных систем информационной безопасности. Были проведены экспериментальные исследования, подтвердившие достоверность полученных результатов и корректность работы системы.

Ключевые слова: динамическая аутентификация, биометрия, одноклассовая классификация, градиентный бустинг, динамическое удаление локальных выбросов, одноклассовый метод опорных векторов, нейронные сети, динамическое изменение уровня доверия.

1. Введение. В настоящее время неотъемлемой частью различных сфер деятельности человека стало использование информационных систем. Огромное количество информации ограниченного доступа переносится, хранится и обрабатывается в информационных системах, что формирует потребность в обеспечении их надежной защиты.

Люди используют механизмы контроля доступа, такие как пароль, магнитные карты или биометрию для защиты от несанкционированного доступа другого человека. Это означает, что пользователь должен предоставить подтверждение своей личности при запуске или разблокировке системы. Контроль доступа к персональному компьютеру обычно реализуется как единоразовое подтверждение личности во время первичной авторизации. Авторизация — это процедура предоставления субъекту определенных прав доступа к ресурсам системы после успешного прохождения им процедуры аутентификации. Предполагается, что в течение всего сеанса в системе будет находиться только авторизованный (легитимный) пользователь. Однако во многих случаях люди оставляют компьютер без присмотра, временно покидая свое рабочее место, и любой человек может получить доступ к тем же источникам данных с теми же правами, что и легитимный пользователь.

Защита информации в информационных системах обеспечивается созданием комплексной системы защиты, одной из главных составляющих которой являются методы защиты от несанкционированного доступа [1, 2]. Основой программно-технических средств защиты от несанкционированного доступа являются процедуры идентификации и аутентификации пользователей. Идентификатором в таком случае служит уникальный признак объекта, позволяющий отличить его от других объектов. А под процедурой аутентификации подразумевается процесс проверки принадлежности субъекту доступа предъявленного им идентификатора. Используемые в существующих

¹ Факультет ВМК МГУ, студ., e-mail: berezniker@mail.ru

² Факультет ВМК МГУ, асс., к.ф.-м.н., e-mail: mkazachuk@cs.msu.ru

³ Факультет ВМК МГУ, проф., д.ф.-м.н., e-mail: mash@cs.msu.ru

⁴ Факультет ВМК МГУ, доц., к.ф.-м.н., e-mail: michael@cs.msu.ru

⁵ Факультет ВМК МГУ, м.н.с., e-mail: ivan@jaffar.cs.msu.ru

системах идентификаторы — секретные знания (пароль, ключ и т.д.), а также физические объекты, принадлежащие пользователю (флеш-накопитель, магнитная карта и т.д.), могут быть с легкостью украдены или скомпрометированы. Поэтому на смену им приходят системы аутентификации, использующие биометрические данные пользователей — уникальные биологические и физиологические характеристики, позволяющие установить личность человека. В отличие от рассмотренных ранее идентификаторов, биометрические образцы невозможно забыть или потерять и намного тяжелее скомпрометировать.

Существующие в настоящее время методы биометрической аутентификации могут быть основаны на физиологических характеристиках человека, находящихся при нем в течение всей его жизни, или поведенческих характеристиках человека, являющихся характеристиками поведения индивидуума и отличающихся относительной устойчивостью и постоянством проявления. Основным недостатком методов проверки пользователей, основанных на физиологической биометрии, является то, что они требуют наличия аппаратных устройств, таких как датчики отпечатков пальцев, сканеры сетчатки глаза и другие, которые дороги и не всегда доступны. Хотя проверка отпечатков пальцев становится широко распространенной в ноутбуках и смартфонах, она все еще недостаточно популярна и не может быть использована в веб-приложениях. Кроме того, отпечатки пальцев могут быть скопированы. В свою очередь, методы, основанные на поведенческой биометрии, не требуют специального оборудования, так как они используют обычные устройства ввода для сбора биометрических данных, такие как мышь и клавиатура.

В зависимости от принципа осуществления процедуры проверки идентификатора, выделяют статическую (эпизодическая проверка идентификатора пользователя) и динамическую (личность пользователя проверяется постоянно на протяжении всей сессии работы за компьютером) аутентификацию. Очевидно, динамическая аутентификация пользователей является предпочтительней, так как она исключает сценарии, при которых злоумышленник получает доступ к информационной системе после того, как легитимный пользователь пройдет процедуру аутентификации. Однако данный подход затрачивает больше ресурсов компьютера за счет непрерывной работы.

Таким образом, мы видим проблемы отсутствия контроля факта смены пользователя и компрометации идентификаторов, которые мы предлагаем решить использованием биометрических поведенческих характеристик пользователя для динамической аутентификации. В настоящее время динамическая аутентификация пользователей на основе анализа работы с компьютерной мышью является актуальным направлением исследований в сфере компьютерной безопасности. Достоинство данного подхода заключается в простоте внедрения: нужно лишь устройство ввода (компьютерная мышь) и специальное программное обеспечение, позволяющее проводить анализ. Однако существующие в данной области решения обладают рядом недостатков — в частности, контролируемым сбором данных, использованием бинарных классификаторов и переобучением на исходных наборах данных. В свою очередь, мы фокусируемся на независимой от контекста системе динамической аутентификации, которая реагирует на каждое отдельное действие, выполненное пользователем.

Настоящая статья имеет следующую структуру. В п. 2 приведен обзор существующих решений по данной тематике. Пункт 3 посвящен предлагаемым подходам по построению признакового пространства. В п. 4 приведено описание предлагаемых методов построения модели пользователя. Пункт 5 посвящен экспериментальному исследованию работы предложенных методов. В п. 6 описывается архитектура разработанного кроссплатформенного приложения динамической аутентификации пользователей. В п. 7 делаются выводы по предложенным алгоритмам.

2. Обзор существующих решений. Различные подходы к сбору экспериментальных данных в литературе отличаются количеством пользователей, принявших участие в исследовании, методами сбора и размерами собранных в итоге данных. Однако наиболее важным различием являются условия, в которых эти данные были собраны. Так, среда сбора данных может быть контролируемой (пользователь выполняет строго поставленные задачи: например, кликает на появляющиеся на экране объекты или работает только с текстовым форматом данных) или неконтролируемой (пользователь работает в привычной для него обстановке и выполняет повседневные

для своей деятельности задачи). Сбор экспериментальных данных в контролируемой обстановке с конкретно поставленной перед пользователем задачей, возможно даже на конкретном компьютере, имеет серьезные недостатки. В этом случае пользователь будет больше сосредоточен на выполнении задачи, и его поведение не будет соответствовать его нормальному состоянию. По этой причине результаты экспериментов в контролируемой среде нельзя обобщить на реальную обстановку. Однако в большинстве существующих работ по данной тематике рассматривается только контролируемый сбор данных.

Наиболее часто используемыми в научных работах признаками, характеризующими особенности траектории движения мыши, являются кинематические характеристики (перемещение, длина траектории, скорость, ускорение), направление движения, а также кривизна траектории перемещения. Данные признаки рассчитываются по следующим собираемым характеристикам: тип действия (нажатие / отжатие клавиш, движение мыши), состояние кнопок мыши, координаты курсора, временная метка. В качестве метода предобработки признакового пространства зачастую используют нормализацию [3]:

$$\hat{x} = \frac{x - \mathbb{E}x}{\sqrt{\mathbb{D}x}}, \quad (1)$$

где $\mathbb{E}x$ — математическое ожидание наблюдения, $\mathbb{D}x$ — его дисперсия.

Сводная информация по качеству работы существующих решений, рассматривающих работу пользователей в неконтролируемой среде, представлена в табл. 1, где N — число пользователей, принявших участие в эксперименте. Как видно из таблицы, в большинстве работ рассматривается бинарная классификация (обучение с учителем), однако в реальной ситуации зачастую может не быть примеров данных нелегитимного класса, поэтому необходимо использовать алгоритмы одноклассовой классификации. Наилучшее качество распознавания в существующих работах показывают одноклассовая машина опорных векторов и нейронные сети типа автокодировщика.

Таблица 1
Качество работы существующих решений

Работа	N	Набор данных	Модель	Обучение без учителя	ROC AUC
[3]	10	BALABIT	Linear SVC	✗	0.81
[4]	28	Неизвестно	SVM	✗	0.85
[5,6]	14	BALABIT+TWOS	2D-CNN	✗	0.88
[7]	25	Неизвестно	Autoencoder	✓	0.91
[8]	50	Неизвестно	One-Class SVM	✓	0.93

3. Построение признакового пространства. В ходе предварительного анализа данных, характеризующих динамику работы пользователей с мышью, были обнаружены следующие особенности: полностью дублирующиеся записи, дубликаты временных меток, множественные дубликаты положения мыши (зацикливание), сверхбольшие значения координат.

Все данные особенности были устранины на этапе предобработки данных:

- дублирующиеся записи были удалены, так как они могли быть следствием сбоев в системе сбора данных на устройствах пользователей, что может привести к некорректному вычислению признаковых характеристик;
- дубликаты временных меток были заменены средним значением соседних меток для обхода ситуации деления на 0 в признаках, связанных со временем:

$$\text{timestamp}_i = \frac{\text{timestamp}_{i-1} + \text{timestamp}_{i+1}}{2};$$

- множественные дубликаты положения мыши (фиксированные координаты курсора (x, y) при изменяющемся времени t) были оставлены, так как такая запись может обозначать паузу в работе пользователя и должна учитываться моделью-классификатором;
- сверхбольшие значения координат были заменены значением верхней границы интерквантильного анализа. Интерквантильный размах (IQR) — это разность между 75 и 25 квантилем. Эмпирический метод заключается в том, что наблюдения, лежащие за пределами верхней и нижней границы, являются выбросами.

Далее, последовательности собранных событий работы пользователя с мышью — кортежей вида \langle тип действия, состояние кнопок мыши, координаты курсора, временная метка \rangle разделялись на сегменты — временные окна, и по каждому временному окну строился вектор признаков, содержащий статистическую информацию о содержащихся в нем событиях. Для построения признакового пространства было предложено использовать конкатенацию наборов признаков из работ [8,9]. Данные признаки описывают особенности траектории движения мыши. Получившееся признаковое пространство имеет размерность 78 признаков на вектор, в него входят следующие характеристики:

- скорость, ускорение, перемещение, длина траектории, а также их дискретизированные, минимальные, максимальные характеристики, среднее значение и стандартное отклонение;
- направление движения (всего рассматривалось восемь направлений);
- средняя кривизна траектории движения: $\frac{1}{n} \sum_{i=0}^n \frac{\angle P(x_i, y_i)P(0, 0)P(x_i, 0)}{\sqrt{x_i^2 + y_i^2}}$, где $P = (x_i, y_i)$ — координаты положения курсора;
- траектория центра масс: $TCM = \frac{1}{S_{n-1}} \sum_{i=1}^{n-1} t_{i+1} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}$, где t_i — временная метка, S_{n-1} — длина пути;
- коэффициент рассеивания: $SC = \frac{1}{S_{n-1}} \sum_{i=1}^{n-1} t_{i+1}^2 \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} - TCM^2$.

Для обработки признакового пространства нами было предложено использовать нормализацию и отбор наиболее важных признаков при помощи градиентного бустинга [10]. Обработка признаков путем One-Hot Encoding кодирования и дискретизации по квантилям [11] в данной работе не рассматривается, поскольку на предварительном этапе экспериментальных исследований использование данных методов не повысило качества распознавания.

Бустинг — это процедура последовательного построения композиции алгоритмов машинного обучения, когда каждый следующий алгоритм стремится компенсировать недостатки композиции всех предыдущих алгоритмов. Представим функцию f классификации в виде композиции T функций, так что каждая из последующих функций минимизирует остатки от предыдущей. Используя метод градиентного спуска, будем модифицировать модель (так как модификация f на основе отклонений аналогична модификации f на основе отрицательного градиента).

Тогда алгоритм градиентного бустинга может быть описан следующими этапами.

- Построение исходной модели $f(x)$.
- В цикле от $t = 1 \dots T$ (по “глубине” композиции):
 - вычисление антиградиента;
 - построение регрессии h_t по антиградиенту;
 - модификация модели: $f(x) \leftarrow f(x) + \rho_t \cdot h_t(x)$, где ρ_t — градиентный шаг.

Преимущество использования ансамблей деревьев решений, таких как градиентный бустинг, заключается в том, что они могут предоставлять оценку важности признаков из обученной модели. Как правило, важность обеспечивает оценку, которая указывает, насколько полезным был каждый признак при построении деревьев решений в модели. Чем больше атрибут используется для принятия ключевых решений, тем выше его относительная важность. Важность рассчитывается для отдельного дерева решений, затем значения характеристик усредняются по всем деревьям решений в модели. Данный метод требует наличия размеченных данных. Поэтому он был использован только для снижения размерности и фильтрации признакового пространства на этапе проведения экспериментов.

4. Построение модели пользователя. Рассматриваемая в рамках данной работы задача относится к классу задач поиска аномалий в данных (обучение без учителя). Аномальным называется объект или событие в выборке, чьи признаки или их комбинации не соответствуют зависимостям, характерным для остальных объектов или событий в данной выборке. Для решения данной задачи нами были рассмотрены следующие алгоритмы одноклассовой классификации: одноклассовый метод опорных векторов (Support Vector Clustering, SVC), изоляционный лес (Isolation Forest), эллипсоидальная аппроксимация данных (Elliptic Envelope).

Поскольку сама обучающая выборка также может содержать в себе аномалии, мы предлагаем использовать технологию локального уровня выбросов (Local Outlier Factor, [12]) для предварительной очистки выборки от аномальных наблюдений. Данный метод обнаруживает выбросы (аномальные наблюдения) на основе локальной плотности точек, которые являются выбросами по отношению к их локальной окрестности, а не по отношению к глобальному распределению данных. Чем выше значение LOF для наблюдения, тем более аномальным оно является. Точка считается аномальной, если ее локальная плотность значительно отличается от плотности соседних точек. Локальный уровень выбросов LOF в точке P определяется как:

$$\text{LOF}(P) = \sum_1 (\text{расстояние до соседей}) \cdot \sum_2 (\text{локальная плотность соседей}). \quad (2)$$

Таким образом, LOF в точке P может принимать:

- большое значение, если точка P находится далеко от своих соседей и ее соседи имеют высокую локальную плотность (т.е. они близки к своим соседям);
- среднее значение, если P находится далеко от своих соседей и ее соседи имеют малую локальную плотность;
- маленькое значение, если P находится близко к своим соседям и ее соседи имеют малую локальную плотность.

Метапараметрами данного алгоритма являются $n_neighbors$ — количество соседей, а также $contamination$ — доля аномалий в выборке.

В методе SVC [8] объекты из исходного множества неявно отображаются с помощью потенциальной функции в пространство характеристик высокой размерности, где далее происходит поиск гиперсферы минимального радиуса, содержащей внутри “основную часть” образов объектов из исходного множества. Аномалиями считаются объекты, чей образ лежит за пределами найденной гиперсферы. Таким образом, в данном методе решается следующая задача оптимизации:

$$\min_{\xi \in \mathbb{R}^N, R \in \mathbb{R}, a \in H} \left[R^2 + \frac{1}{\nu N} \sum_{i=1}^N \xi_i \right], \quad (3)$$

$$\|\varphi(x_i) - a\|_H^2 \leq R^2 + \xi_i \quad \forall i \in [1, N],$$

где $\varphi(x_i)$ — образ исходного объекта x_i в пространстве характеристик высокой размерности H , R — радиус построенной гиперсферы, a — центр гиперсферы, N — число объектов в обучающей выборке X , $0 < \nu \leq 1$ — предопределенный процент аномалий, ξ_i — дополнительные переменные.

Настраиваемыми параметрами данного алгоритма являются *kernel* — тип потенциальной функции и ν — отношение ожидаемого числа аномалий к общему числу объектов рассматриваемой выборки.

Отметим, что другой разновидностью одноклассового метода опорных векторов является метод One-Class SVM. Он строит в пространстве характеристик высокой размерности не гиперсферу, а гиперплоскость, отделяющую нормальные данные от аномальных. Заметим, что при использовании радиально-базисной потенциальной функции (*rbf*) результаты работы методов SVC и One-Class SVM идентичны.

Изоляционный лес [13], как и любой другой метод ансамбля деревьев, построен на основе деревьев решений. Алгоритм обучения строит ансамбль изоляционных деревьев на основе рекурсивной и рандомизированной процедуры структурированного разбиения: сначала случайным образом выбирается объект, а затем выбирается случайное значение между минимальным и максимальным значением выбранного объекта. Аномалий в данных немного и зачастую они находятся дальше от обычных наблюдений в пространстве признаков, поэтому при использовании такого случайного разбиения они должны быть идентифицированы ближе к корню дерева. В случае аномальных наблюдений необходимо меньше расщеплений. Для принятия решения об аномальности наблюдения данный алгоритм использует следующую функцию:

$$S(x, n) = 2^{-\frac{\mathbb{E}(h(x))}{c(n)}}, \quad (4)$$

где $\mathbb{E}(h(x))$ — средняя длина пути наблюдения x , $c(n)$ — средняя длина пути неудачного поиска в бинарном дереве поиска, n — количество внешних узлов.

Каждое наблюдение получает индекс аномальности, на основе которого принимается решение. Оценка, близкая к 1, указывает на аномальность наблюдения. Оценка, близкая к 0, указывает на нормальное поведение наблюдения. Данный алгоритм обладает следующими метапараметрами: *n_estimators* — число деревьев, *max_samples* — объем выборки для построения одного дерева, *contamination* — доля аномалий в выборке. Он устойчив к проклятию размерности и обладает высокой эффективностью.

Эллипсоидальная аппроксимация данных [14] — ковариационная оценка, предполагающая, что данные имеют распределение Гаусса. Ограничивающий контур имеет эллиптическую форму. Для оценки размера и формы эллипса используется алгоритм FAST-Minimum Covariance Determinate. Данный алгоритм выбирает неперекрывающиеся подвыборки данных и вычисляет среднее значение μ и ковариационную матрицу C в признаковом пространстве для каждой подвыборки. Расстояние Махalanобиса dMH вычисляется для каждого многомерного вектора данных (признака) x в каждой подвыборке, по формуле

$$dMH = \sqrt{(x - \mu)^T C (x - \mu)}. \quad (5)$$

Затем данные упорядочиваются по возрастанию значений dMH . Эта процедура повторяется до сходимости определителя матрицы ковариации. Ковариационная матрица с наименьшим определителем из всех подвыборок образует эллипс, который охватывает часть исходных данных. Данные в пределах поверхности эллипса считаются нормальными, а вне эллипса — аномальными. Метапараметром данного алгоритма является *contamination* — доля выбросов в выборке. Однако, данный метод успешно работает только на нормально распределенных одномодальных данных.

Автокодировщик (Autoencoder) [7] — это нейронная сеть, которая восстанавливает входной сигнал на выходе. С математической точки зрения каждый нейрон в нейронной сети представляет собой параметризованную функцию $f_{\omega_1, \dots, \omega_m}(x_1, \dots, x_n, b)$, возвращающую единственное значение — величину выходного сигнала. Параметры $\omega_0, \dots, \omega_m$ функции $f_{\omega_1, \dots, \omega_m}(x_1, \dots, x_n, b)$ называются синаптическими весами нейрона. Чаще всего для агрегации входов используется линейная комбинация входов нейрона x_1, \dots, x_n и некоторого смещения b , также являющегося параметром функции $f_{\omega_1, \dots, \omega_m}(x_1, \dots, x_n, b)$. Агрегированное значение входов нейрона подается на вход функции активации нейрона, которая непосредственно вычисляет величину выходного сигнала.

Связанные между собой нейроны образуют нейронную сеть. В том случае, когда нейронная сеть состоит из большого количества нейронов, вводится также понятие слоя нейронной сети и все слои разделяются на три типа: входной слой, скрытый слой, выходной слой. Вектор входных значений, именуемый также вектором признаков, подается на входной слой нейронной сети и в результате работы последней на выходном слое появляется вектор выходных значений, называемый также предсказанием нейронной сети.

Автокодировщик состоит из двух частей: кодировщика (encoder), который кодирует данные в свое внутреннее представление, и декодировщика (decoder), который восстанавливает исходный вектор. Обычно автокодировщиков ограничивают в размерности скрытых слоев (они меньше, чем размерность сигнала на входе). Нами было предложено несколько архитектур нейронных сетей для решения поставленной задачи. Для борьбы с переобучением мы использовали L_2 -регуляризацию и Dropout-слои.

Процедура L_2 -регуляризации выполняется посредством наложения штрафов на веса с наибольшими значениями, минимизируя их L_2 -норму с использованием коэффициента регуляризации λ . Для каждого веса ω мы прибавляем к целевой функции \mathcal{L} слагаемое:

$$\frac{\lambda}{2} \omega^T \omega = \frac{\lambda}{2} \|\omega\|_2 = \frac{\lambda}{2} \sum_{i=1}^n \omega_i^2. \quad (6)$$

Идея Dropout состоит в том, что вместо обучения одной нейронной сети мы обучим несколько и сделаем из них ансамбль с усреднением полученных результатов. Такие сети получаются с помощью исключения нейронов с вероятностью p . Исключение нейрона означает, что при любых входных данных он возвращает 0.

На рис. 1 представлена предложенная архитектура сети автокодировщика с семью скрытыми слоями. Перед каждым полно связанным слоем используется функция активации ELU (7):

$$\text{ELU}(x) = \begin{cases} x, & x > 0, \\ e^x - 1, & x \leq 0. \end{cases} \quad (7)$$

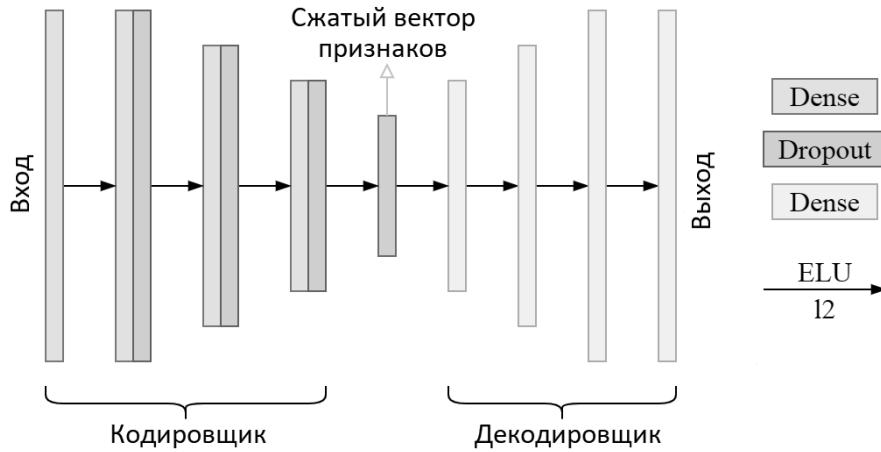


Рис. 1. Предложенная архитектура полно связного автокодировщика

В качестве функции потерь мы использовали функцию Хьюберта, так как она устойчива к выбросам:

$$L_\delta(x) = \begin{cases} \frac{1}{2}x^2, & |x| \leq \delta, \\ \delta(|x| - \frac{1}{2}\delta), & |x| > \delta. \end{cases} \quad (8)$$

Для принятия решения о легитимности пользователя будем оценивать значение метрики MSE (9) между вектором признаков на входе (y^{true}) и на выходе (y^{pred}) автокодировщика. Чем лучше

сеть восстанавливает вектор, тем меньше ошибка, а значит будем считать, что уверенность модели в легитимности текущего пользователя больше:

$$\text{MSE}(y^{true}, y^{pred}) = \sqrt{\sum_{i=1}^n (y_i^{true} - y_i^{pred})^2}. \quad (9)$$

Также была предложена нейронная сеть с архитектурой сверточного автокодировщика (CNN-Autoencoder), представленная на рис. 2. Основной особенностью полносверточных нейросетей является относительно небольшое число параметров за счет использования сверточных слоев, что позволяет конструировать и обучать более сложные архитектуры для повышения качества решения поставленной задачи. В данной архитектуре мы также использовали L_2 -регуляризацию и Dropout-слои для борьбы с переобучением. В качестве функции активации была выбрана функция RELU (10):

$$\text{ReLU}(x) = \max(0, x). \quad (10)$$

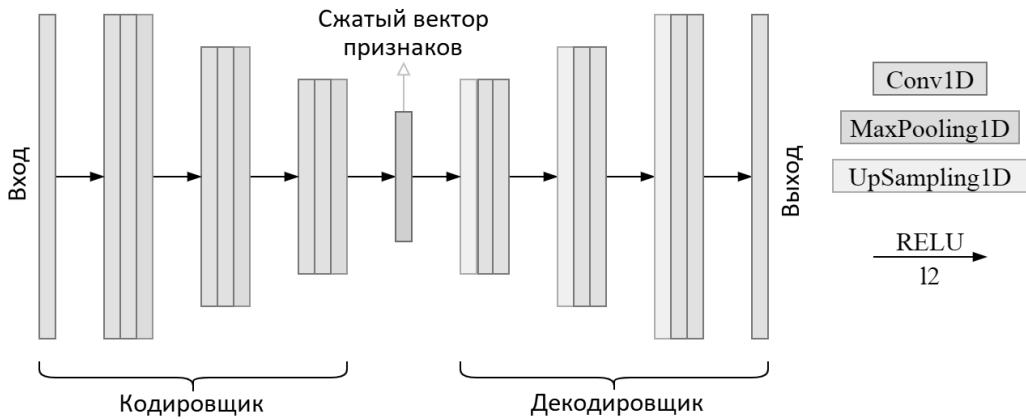


Рис. 2. Предложенная архитектура сверточного автокодировщика

В качестве динамической составляющей системы аутентификации мы предлагаем рассмотреть модель уровня доверия (Trust Model, [8]). Основная идея данной технологии заключается в том, что доверие (или уверенность) системы в подлинности текущего пользователя зависит от его отклонений от нормального состояния. Если текущее действие выполняется в соответствии с шаблоном, хранящимся в профиле легитимного пользователя, то доверие системы к подлинности текущего пользователя будет увеличено (вознаграждение). Если между поведением подлинного и текущего пользователя будет заметное различие, то доверие системы к этому пользователю будет уменьшаться (штраф). Количество изменений уровня доверия может быть фиксированным или переменным, величина штрафа или вознаграждения задается дельта-функцией.

Ни один человек не сможет всегда вести себя одинаково. Для легитимного пользователя это означает, что его поведение также иногда будет отклоняться от его нормального (шаблонного) поведения, что повлечет к снижению уровня доверия. Тем не менее, большинство действий легитимного пользователя будут близки к его нормальному поведению, т.е. будут приводить к повышению уровня доверия. В целом это приведет к высокому уровню доверия системы. Для злоумышленника, однако, верно обратное. Лишь в некоторых случаях он сможет вести себя как подлинный пользователь, увеличивая свой уровень доверия, но большинство его действий будут вести к снижению доверия из-за большого отклонения от поведения легитимного пользователя, что в итоге приведет к общему снижению доверия со временем и блокировке.

Алгоритм Trust Model

Используются следующие понятия:

score_i — отклик базового классификатора для i -го действия;

$Trust_i$ — уверенность алгоритма для i -го действия;

$Trust_{lockout}$ — уровень блокировки;

A — порог для штрафа или награды;

B — ширина сигмоиды, $B > 0$;

C — максимальная награда, $C > 0$;

D — максимальный штраф, $D > 0$.

До тех пор пока $Trust_i > Trust_{lockout}$ выполнять

$score_i = model.score(data_i)$ — расчет отклика $score_i$ классификатора $model$ на новой порции данных $data_i$;

$$\Delta_T(score_i) = \min \left\{ -D + \left(\frac{D \times \left(1 + \frac{1}{C} \right)}{\frac{1}{C} + \exp \left(-\frac{score_i - A}{B} \right)} \right), C \right\};$$

$$Trust_i = \min \{ \max \{ Trust_{i-1} + \Delta_T(score_i), 0 \}, 100 \};$$

Конец цикла

Заметим, что дельта-функция $\Delta_T(score_i)$ представляет собой сигмоиду с параметрами: A, B, C, D . Для плавного изменения доверия к пользователю были подобраны следующие значения: $A = 0.0$, $B = 0.25$, $C = 1.0$, $D = 1.0$. График данной функции представлен на рис. 3.

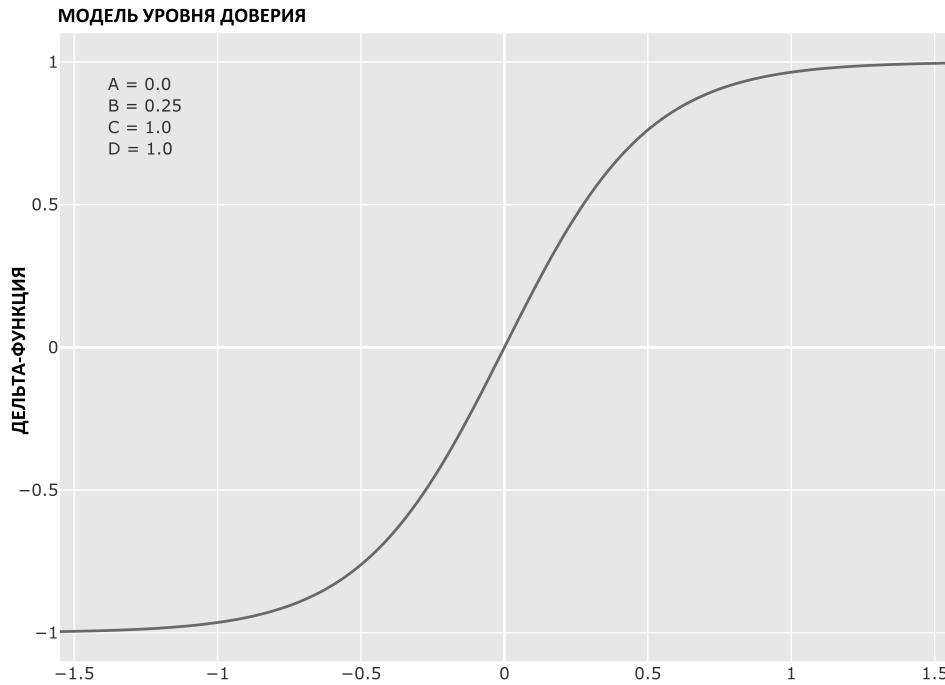


Рис. 3. Дельта-функция динамического изменения уровня доверия

5. Экспериментальное исследование. Нами было найдено четыре открытых набора данных, собранных в неконтролируемой среде: BALABIT, использующийся в работах [3,5,6], TWOS (см. [5,6]), DFL [15] и CHAOSHEN [16]. В остальных исследовательских публикациях используются собственные или недоступные наборы данных. Также нами самостоятельно был собран собственный набор данных DATAIT. Характеристики данных наборов представлены в табл. 2.

Каждый набор данных был разбит на тренировочную и тестовую части, 75% и 25% от исходных данных соответственно. Каждая часть содержит в себе записи об авторизованных сессиях всех пользователей из множества $\mathfrak{U} = \{U_1, \dots, U_j, \dots, U_q\}$. Работа пользователя разбита на сессии разной продолжительности $U_j = \{S_1, \dots, S_i, \dots, S_m\}$, где каждая сессия $S_i \in U_j$, $i = \overline{1, m}$, $j = \overline{1, q}$, содержит записи вида $(time, x_{pos}, y_{pos})$, где $time$ — временная метка, (x_{pos}, y_{pos}) — координаты позиций курсора. Далее каждую сессию мы разбиваем на сегменты с ограничением по временному порогу сверху и по минимальному количеству действий в этот промежуток времени снизу. По

Таблица 2

Наборы данных для экспериментальных исследований

Набор данных	Число пользователей	Число событий на пользователя	Время записи на пользователя
BALABIT	10	700 000	43 ч
DATAIT	20	1 150 000	20 ч
TWOS	4	400 000	10 ч
DFL	21	7 400 000	100 ч
CHAOSEN	50	200 000	4 ч

полученному сегменту строится один вектор признаков. В результате анализа наборов данных было получено, что оптимально проводить разбиение на сегменты продолжительностью 3–5 секунд: данные сегменты будут содержать достаточное количество действий пользователя для уникальности признаков, также это позволит проводить частую проверку пользователя. Количество векторов признаков в обучающей выборке также будет достаточно высоким (порядка 3500 векторов для каждого пользователя). В рамках экспериментов все найденные наборы данных были объединены, т. е. проводилось тестирование “один против всех” по каждому из 105 пользователей результирующего набора.

В результате экспериментального исследования было получено, что нормализация признакового пространства способствовала ускорению и увеличению качества работы методов в среднем на 0.05 по метрике ROC AUC. Дополнительное улучшение распознавания (на 0.02 ROC AUC) дал отбор признаков на основе градиентного бустинга. Использование динамического обнаружения неконтролируемых локальных выбросов на основе технологии Local Outlier Factor перед обучением модели позволило увеличить качество работы методов в среднем еще на 0.02. Подбор метапараметров алгоритмов машинного обучения осуществлялся жадным перебором по сетке. В результате были выбраны следующие их значения (см. табл. 3).

Таблица 3

Метапараметры алгоритмов машинного обучения

Метод машинного обучения	Метапараметры алгоритма
SVC	kernel = “rbf” $\nu = 0.1$
Isolation Forest	n_estimators = 10 max_samples = 0.7 contamination = 0.1
Local Outlier Factor	n_neighbors = 10 contamination = 0.1
Elliptic Envelope	contamination = 0.1

Окончательные результаты экспериментов представлены в табл. 4. В качестве метрики использовались среднее (mean), медианное (median) значение площади под ROC-кривой (ROC AUC), а также межквартильный размах (IQR). Достоинством ROC AUC является то, что его значение не зависит от выбранного порога, который определяет, насколько близким к эталону должен быть вывод алгоритма классификации, чтобы его можно было считать совпадающим с эталоном. Лучший результат продемонстрировала следующая комбинация методов:

- нормализация признакового пространства;
- градиентный бустинг для выявления значимых признаков;

- динамическое удаление локальных выбросов в обучающей выборке на основе технологии Local Outlier Factor;
- одноклассовая машина опорных векторов SVC в качестве классификатора.

Таблица 4
Результаты экспериментов

	Нормализация	Градиентный бустинг	Отлов выбросов	Среднее ROC AUC	Медианное ROC AUC ± IQR
SVC	–	–	–	0.732	0.750 ± 0.032
	–	–	+	0.752	0.770 ± 0.031
	–	+	–	0.752	0.770 ± 0.029
	–	+	+	0.772	0.790 ± 0.045
	+	–	–	0.782	0.800 ± 0.043
	+	–	+	0.802	0.820 ± 0.021
	+	+	–	0.802	0.820 ± 0.025
	+	+	+	0.822	0.840 ± 0.038
CNN-Autoencoder	–	–	–	0.727	0.747 ± 0.031
	–	–	+	0.747	0.767 ± 0.027
	–	+	–	0.747	0.767 ± 0.021
	–	+	+	0.767	0.787 ± 0.041
	+	–	–	0.777	0.797 ± 0.026
	+	–	+	0.797	0.817 ± 0.045
	+	+	–	0.797	0.817 ± 0.042
	+	+	+	0.817	0.837 ± 0.034
Autoencoder	+	+	+	0.732	0.749 ± 0.061
Isolation Forest	+	+	+	0.655	0.673 ± 0.072
Elliptic Envelope	+	+	+	0.542	0.543 ± 0.101

6. Кроссплатформенное приложение динамической аутентификации пользователей. На основе результатов проведенных исследований была спроектирована архитектура, представленная на рис. 4, и реализовано кроссплатформенное приложение для динамической аутентификации пользователей на основе анализа работы с компьютерной мышью. Несмотря на то, что предложенная архитектура сверточного автокодировщика почти не уступила по качеству распознавания алгоритму SVC, в качестве базового классификатора была выбрана одноклассовая машина опорных векторов, поскольку для обучения этой модели нужно меньше времени и вычислительных ресурсов. Для реализации системы был выбран язык программирования Python 3. Корректность работы системы была протестирована на операционных системах Windows 8.1, MacOS Catalina 10.15.6, Ubuntu 20.04 LTS.

После прохождения процедуры авторизации пользователь получает свой уникальный идентификатор — UID (UserID), который характеризует его в системе и сопоставляется для него информацию в базе данных (данные динамики работы с компьютерной мышью и модель-классификатор).

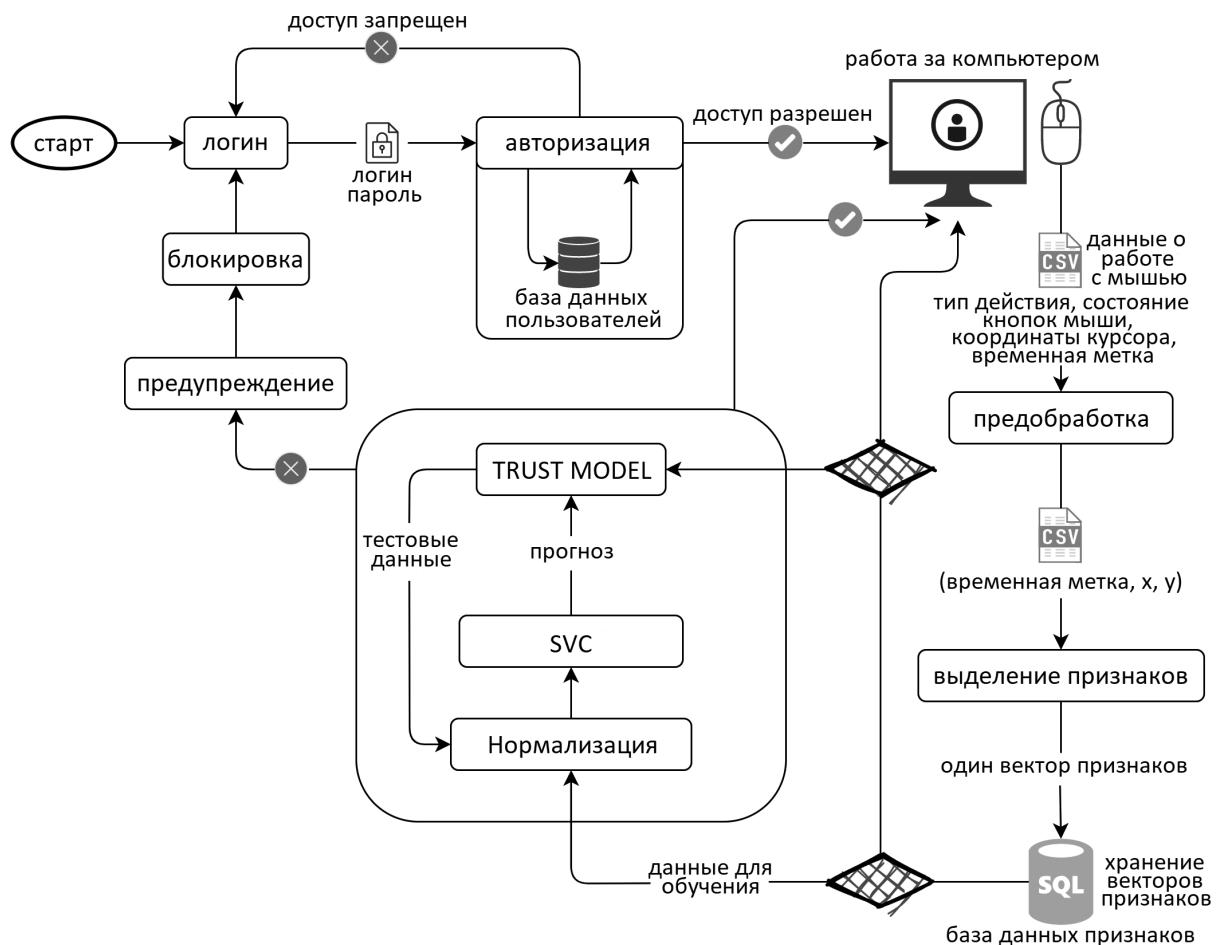


Рис. 4. Архитектура кроссплатформенного приложения

В качестве инструмента для работы с базой данных используется встроенная в язык программирования библиотека SQLite. После прохождения процедуры авторизации запускается основной цикл по сбору, обработке, хранению данных и динамическому принятию решения о легитимности текущего пользователя. В течение 1 часа активной работы пользователя с компьютерной мышью собираются данные для обучения модели. Обученная модель сохраняется для дальнейшего использования. При наличии обученной модели каждые 3 секунды происходит проверка подлинности и изменение значения уровня доверия Trust Model на основе предсказаний алгоритма SVC. В случае обнаружения вторжения в панель управления устройства будет отправлено уведомление с предупреждением и через 5 секунд произойдет блокировка всей системы, после чего потребуется повторная авторизация в системе и приложении.

Также был реализован тестовый режим для возможности имитации работы пользователя путем подачи событий из тестовых наборов данных и замера показателей эффективности по временным окнам:

- mean FRR = 0.089, median FRR \pm IQR = 0.086 ± 0.002 ;
- mean FAR = 0.071, median FAR \pm IQR = 0.070 ± 0.001 ;
- mean ANIA = 102 секунд, median ANIA \pm IQR = 111 ± 36 секунд;
- mean ANGA = 118 минут, median ANGA \pm IQR = 117 ± 5 минут.

Порог для вычисления FRR (ошибки 1-го рода) и FAR (ошибки 2-го рода) подбирается оптимальным образом для минимизации EER средствами библиотеки языка Python – scikit-learn. Под

ошибкой первого рода понимается доля случаев, когда биометрическая система не предоставляет доступ легитимному пользователю. Под ошибкой второго рода — доля случаев, когда система предоставляет доступ неуполномоченному лицу. EER — это равный коэффициент ошибок, он дает возможность сравнивать системы: меньшее значение EER означает, что можно настроить систему так, чтобы частота ошибок как 1-го рода, так и 2-го рода была меньше. Также были замерены значения показателей ANIA (Average Number of Imposter Actions, среднее количество действий злоумышленника, — показывает, сколько действий успеет совершить злоумышленник до того момента, как он будет заблокирован системой) и ANGA (Average Number of Genuine Actions, среднее количество легитимных действий, — характеризует количество действий авторизованного пользователя до его ошибочной блокировки системой). Отметим, что полученные показатели ANIA и ANGA свидетельствуют о том, что разработанную систему можно применять на практике.

7. Заключение. Динамическая аутентификация пользователей на основе анализа работы с компьютерной мышью является одним из наиболее перспективных направлений развития современных методов аутентификации. Она позволяет выявлять внутренние атаки в информационные системы в кратчайшие сроки и при минимальных затратах на дополнительное оборудование. В данной работе были предложены подходы к предобработке данных и построению признакового пространства, включающие в себя фильтрацию собранных данных, градиентный бустинг для отбора наиболее значимых признаков, динамическое удаление локальных выбросов и нормализацию получившегося признакового пространства. Комбинация данных подходов способствовала улучшению качества распознавания. Предложенный метод динамической аутентификации пользователей на основе полностью сверточной нейронной сети типа автокодировщика по результатам экспериментальных исследований показал сопоставимые по точности результаты с методом, основанным на использовании алгоритма SVC. Решение о блокировке текущего пользователя принимается на основе динамически изменяющегося уровня доверия пользователя алгоритмом Trust Model. На основе разработанных алгоритмов была спроектирована и реализована кросплатформенная система динамической аутентификации пользователей, обладающая показателями эффективности, позволяющими использовать ее для построения перспективных современных систем информационной безопасности, включающих в себя средства анализа динамики работы пользователей с компьютерной мышью. В дальнейшем предлагается исследовать стабильность разработанных подходов при смене используемого оборудования.

СПИСОК ЛИТЕРАТУРЫ

1. Jain A., Ross A., Prabhakar S. An introduction to biometric recognition //IEEE Transactions on circuits and systems for video technology. 2004. **14**. N 1. P. 4–20.
2. Wayman J. et al. An introduction to biometric authentication systems //Biometric Systems. London: Springer, 2005. P. 1–20.
3. Tan Y., Binder A., Roy A. Insights from curve fitting models in mouse dynamics authentication systems //IEEE Conference on Application, Information and Network Security. IEEE, 2017. P. 42–47.
4. Khalifa A. A. et al. Comparison between mixed binary classification and voting technique for active user authentication using mouse dynamics //2015 International Conference on Computing, Control, Networking, Electronics and Embedded Systems Engineering (ICCNEEE). IEEE, 2015. P. 281–286.
5. Chong P., Elovinci Y., Binder A. User authentication based on mouse dynamics using deep neural networks: a comprehensive study //IEEE Transactions on Information Forensics and Security. 2019. P. 1086–1101.
6. Chong P. et al. Mouse authentication without the temporal aspect — what does a 2D-CNN learn? //IEEE Security and Privacy Workshops (SPW). IEEE, 2018. P. 15–21.
7. Antal M., Fejer N. Mouse dynamics based user recognition using deep learning //Acta Universitatis Sapientiae, Informatica. 2020. **12**. N 1. P. 39–50.
8. Mondal S., Bouras P. A study on continuous authentication using a combination of keystroke and mouse biometrics //Neurocomputing. 2017. **230**. P. 1–22.
9. Fehér C. et al. User identity verification via mouse dynamics //Information Sciences. 2012. **201**. P. 19–36.
10. Friedman J. H. Greedy function approximation: a gradient boosting machine //Annals of Statistics. 2001. **29**. N 5. P. 1189–1232.

11. Kazachuk M., Kovalchuk A., Mashchkin I., Orpanen I., Petrovskiy M., Popov I., Zakliakov R. One-class models for continuous authentication based on keystroke dynamics //International Conference on Intelligent Data Engineering and Automated Learning. Cham: Springer, 2016. P. 416–425.
12. Breunig M. M. et al. LOF: identifying density-based local outliers //Proceedings of the 2000 ACM SIGMOD international conference on Management of data. N.Y., USA: ACM, 2000. P. 93–104.
13. Liu F.T., Ting K.M., Zhou Z.H. Isolation forest //Eighth IEEE International Conference on Data Mining. IEEE, 2008. P. 413–422.
14. Hoyal B. et al. Anomaly detection for machine learning redshifts applied to SDSS galaxies //Monthly Notices of the Royal Astronomical Society. 2015. **452**. N 4. P. 4183–4194.
15. Antal M., Denes-Fazakas L. User verification based on mouse dynamics: a comparison of public data sets //IEEE 13th International Symposium on Applied Computational Intelligence and Informatics (SACI). IEEE, 2019. P. 143–148.
16. Shen C., Cai Z., Guan X. Continuous authentication for mouse dynamics: A pattern-growth approach //IFIP International Conference on Dependable Systems and Networks (DSN 2012). 2012. P. 1–12.

Поступила в редакцию 09.08.21

После доработки 31.08.21

Принята к публикации 31.08.21