

ПРИМЕНЕНИЕ МЕТОДОВ МАШИННОГО ОБУЧЕНИЯ ДЛЯ ПРОТИВОДЕЙСТВИЯ DGA-УГРОЗАМ В СИСТЕМЕ УПРАВЛЕНИЯ КАЧЕСТВОМ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ ПРЕДПРИЯТИЙ

Королев И.А., магистр, Финансовый университет при Правительстве РФ, г. Москва

Кобелев Е.А., магистр, Финансовый университет при Правительстве РФ, г. Москва

Булгакова Е.В., доцент, канд. юрид. наук, доцент Московского университета МВД России имени В.Я. Кикотя, г. Москва

Кубанков А.Н., профессор, д-р воен. наук, главный научный сотрудник, Российский институт стандартизации

В данной статье определяется понятие технологии DGA, проводится анализ существующих решений для распознавания её применения. Формируется набор моделей машинного обучения, подходящих для определения зараженных доменов, на основе сравнения и обучения которых разрабатывается нейронная сеть для автоматической классификации данных. Описаны этапы подготовки данных, построения бинарного классификатора и оценки его эффективности. Проведено сравнение моделей Random Forest, Gradient Boosting и Logistic Regression по метрикам точности, полноты, F1-меры и ROC AUC. Результаты показали высокую эффективность при комбинировании методов машинного обучения внутри нейронной сети, особенно с использованием Gradient Boosting. Разработанный алгоритм применим в реальном времени и может быть интегрирован в информационные системы для защиты сетевой инфраструктуры.

Ключевые слова: DGA, домен, хост, ботнет, машинное обучение, нейронная сеть, бинарная классификация.

Цитирование: Королев И.А., Кобелев Е.А., Булгакова Е.В., Кубанков А.Н. Применение методов машинного обучения для противодействия DGA-угрозам в системе управления качеством информационной безопасности предприятий // Информационно-экономические аспекты стандартизации и технического регулирования. 2025. № 4(85). С. 46–55.

1. АКТУАЛЬНОСТЬ, НАУЧНАЯ НОВИЗНА И ПОСТАНОВКА ЗАДАЧИ

Актуальность исследования обусловлена несколькими причинами: увеличение атак с использованием ботнетов, а также участвовавшие случаи появления нового вредоносного ПО, использующего DGA. Создание алгоритма на основе машинного обучения, способного определять вредоносные домены на ранних стадиях атаки, является важной теоретической и практической задачей.

Дополнительно, в контексте современных требований к системам управления качеством информационной безопасности предприятий, разработка подобных решений способствует формированию новой парадигмы цифрового контроля. Такой подход позволяет значительно повысить устойчивость корпоративных ИТ-инфраструктур, автоматизировать процессы мониторинга и повысить точность идентификации угроз, что в совокупности ведет к укреплению системы управления качеством в условиях цифровой трансформации.

Необходимо отметить, что при создании классификатора, способного противостоять DGA, необходимо:

1. Определить оптимальную архитектуру нейронной сети, способной идентифицировать DGA за приемлемый промежуток времени.
2. Обучить нейросеть на подготовленных данных.
3. Проанализировать полученные результаты.

Большинство существующих подходов нацелено на выделение определенных атрибутов у сетевых пакетов (потоков) и формирование на их основе обучающей выборки, которая подается на вход алгоритмам классификации для обучения. Обученные алгоритмы с определенной точностью выявляют тип приложения без использования анализа портов или поиска меток среди полезной нагрузки.

Научная новизна предложенного метода выявления DGA-доменов основывается на том, чтобы помимо проведения сравнительного анализа различных алгоритмов классификации выявить наиболее подходящий под задачи выявления алгоритм DGA и на основе него построить многослойную ней-

ронную сеть (бинарный классификатор). Предполагается, что использование специфичных для бинарной классификации слоев нейросети наравне с выбранной моделью машинного обучения позволит увеличить точность конечного решения.

Задача исследования: разработка с помощью методов машинного обучения алгоритма выявления доменов, сгенерированных алгоритмом DGA, с точностью более 90%. Потери при этом не должны превышать 5% в течение всего процесса обучения алгоритма.

2. ВВЕДЕНИЕ

DGA (Domain Generation Algorithm) – алгоритм, который используется киберпреступниками для генерации большого количества уникальных доменных имен. Чаще всего DGA является временным и применяется для создания ботнетов, а также для распространения вредоносного ПО и прочих злонамеренных действий.

DGA позволяет злоумышленникам обходить бэк-листы и другие системы фильтрации трафика. Помимо этого алгоритм активно используется в качестве командных серверов для управления зараженными устройствами [1].

При управлении ботнетами ключевой проблемой связи между зараженными устройствами и сервером C&C выступает статическая и жесткая IP-адресация, которая может быть легко заблокирована. Таким образом, для целей динамического обновления местоположения C&C сервера были придуманы методы управления ботнетами на основе временных доменных имен.

Кроме этого злоумышленники чаще всего не полагаются на статическое доменное имя из-за угрозы его быстрой блокировки, вместо этого они применяют различные приемы (в том числе и машинного обучения) для быстрой генерации доменных имен.

Д.С. Берман в своей работе [2] указывает, что суть DGA – использование алгоритмов на основе текущего времени или какой-то другой информации по общедоступным хостам. В результате генерируются вредоносные домены, действительные только в течение короткого промежутка времени. Зараженные хосты просматривают все значения-кандидаты, чтобы найти домен, который на самом деле был зарегистрирован владельцем ботнета, что с точки зрения защиты делает этот механизм очень сложным и дорогостоящим для подавления, поскольку изъятие или блокировка тысяч до-

менных имен в день часто являются слишком сложными с административной точки зрения и не масштабируются.

Стоит ожидать, что владельцы ботнетов будут применять новые методы для генерации DGA. Может быть создан новый алгоритм, который специализируется на измерении энтропии или списка слов. Например, создание алгоритма, который использует объединение случайных слов в доменное имя вместо использования случайных букв.

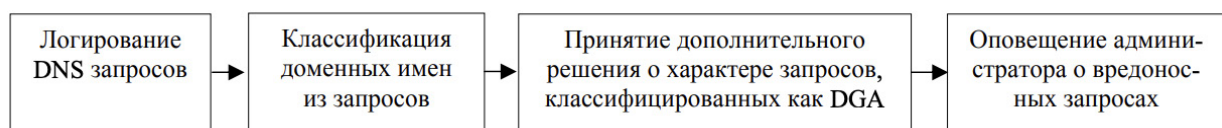
Современные подходы к обеспечению информационной безопасности предприятий все чаще требуют внедрения интеллектуальных инструментов раннего реагирования, что позволяет формировать новую модель системы управления качеством, ориентированную на проактивное противодействие ИБ-угрозам. Одной из таких угроз является DGA, и технологии на базе искусственного интеллекта становятся неотъемлемым компонентом этой модели.

В трудах [3–6], связанных с решением задач классификации трафика, предлагаются методы, которые основаны на машинном обучении, для идентификации DGA и их блокировки для минимизации угроз на сетевую инфраструктуру.

Д.Г. Галиахметов в своей работе рассматривает альтернативное им практическое применение методов машинного обучения на основе Q-обучения для POMDP, в ней показана высокая точность применяемых методов, что доказывает целесообразность применения машинного обучения в задачах идентификации DGA [7].

3. РАЗРАБОТКА НЕЙРОННОЙ СЕТИ ДЛЯ ВЫЯВЛЕНИЯ ДОМЕНОВ, СГЕНЕРИРОВАННЫХ АЛГОРИТМОМ DGA

В качестве инфраструктуры информационной системы рассматривается доменная сеть, под объектом информационной системы понимается любой активный узел сети, способный генерировать запросы от клиента к серверу (далее – DNS-запросы) [8]. Взаимодействие объекта информационной системы с зараженным DGA доменом, заключается, как минимум, в инициировании этим объектом запроса на разрешение домена DGA. Обнаружить взаимодействие объектов информационной системы с DGA доменами возможно путем анализа записей журнала DNS-запросов: локальные записи пересылаются на контроллер домена средствами механизма логирования внутри информационной системы, где и происходит их последующая обработка [9]. Концептуальная схема процесса обнаружения DGA-доменов изображена на схеме:



Классификатору на вход подается множество доменных имен, каждое из которых он должен отнести к одному из двух классов:

- 1) DGA (Domain Generation Algorithm) – клиент-серверный запрос, поступивший с этого домена (имеющий это доменное имя), рассматривается как вредоносный;
- 2) legit/alexa – клиент-серверный запрос, поступивший с этого домена, рассматривается как легитимный.

Технологическую основу классификатора составляет язык программирования Python и набор решений из библиотеки scikit-learn, который позволяет обрабатывать данные, инициализировать и обучать модели машинного обучения и оценивать различные реализации классификатора.

Источником DGA-доменов стал сервис Kaggle, на котором в открытом доступе хранятся первично подготовленные наборы данных для обучения ML-моделей.

Концептуально исследование можно разбить на 6 основных этапов:

1. Подготовка данных (датасетов)
2. Выбор, создание и обучение трех моделей машинного обучения. Для каждой модели проделываем следующие операции:
 - a. Выбор и настройка специфичных для модели параметров, инициализация модели
 - b. Токенизация данных (преобразование значений в числовые)

- c. Обучение модели на обучающей выборке
- d. Обучение модели на тестовой выборке
- e. Визуализация результатов обучения на каждой из выборок
3. Сравнение данных, выбор модели на основе результатов обучения в качестве ядра классификатора
4. Создание и обучение на основе выбранного ядра классификатора двух нейронных сетей с различными слоями. Для каждой нейронной сети выполняем:
 - a. Обучение на обучающей выборке
 - b. Обучение на тестовой выборке
 - c. Визуализация результатов обучения на каждой из выборок
5. Сравнение данных об изменении точности и потерь, выбор нейронной сети на основе результатов обучения в качестве ядра классификатора
6. Подведение итогов исследования, обоснование выбора финального технического решения, формулирование выводов.

Основной датасет, используемый для обучения классификатора – выборка из 160 000 доменных имен. Обучающая выборка составляет 80% от набора данных, тестовая – остальные 20%. Для разбивки данных на части была написана функция train_test_split. Все строки с пропущенными значениями подвергаются удалению с целью избежать проблем при обучении модели, так как это значительно повлияет на точность работы на реальных данных.

	isDGA	domain	host	subclass	second_domain	top_domain
0	1	6xxzsw3sokvg1tc752y1a6p0af	6xxzsw3sokvg1tc752y1a6p0af.com	gameoverdga	6xxzsw3sokvg1tc752y1a6p0af	com
1	1	gibtxwwhbnpxs	gibtxwwhbnpxs.ru	cryptolocker	gibtxwwhbnpxs	ru
2	1	xxmamopyipbfpk	xxmamopyipbfpk.ru	cryptolocker	xxmamopyipbfpk	ru
3	1	zfd5szpi18i85wj9uy13i69rg	zfd5szpi18i85wj9uy13i69rg.net	newgoz	zfd5szpi18i85wj9uy13i69rg	net
4	1	jpqftymiuver	jpqftymiuver.ru	cryptolocker	jpqftymiuver	ru
...	...	...	...	...	...	...
159995	0	manhunt	manhunt.net	alexa	manhunt	net
159996	0	yupptv	yupptv.com	alexa	yupptv	com
159997	0	bbcmundo	bbcmundo.com	legit	bbcmundo	com
159998	0	kanogames	kanogames.com	legit	kanogames	com
159999	0	roadrover	roadrover.cn	alexa	roadrover	cn

159998 rows x 6 columns

Рис. 1. Датасет dga_data.csv

Поскольку задача классификации поставлена как бинарная, а в выборке доменных имен сгенерированных и реальные записи представляют множество подклассов с разными технологиями, реализующими DGA, первоочередная обработка данных заключалась в группировке всех подклассов на два бинарных: newgoz (по названию самого распростра-

ненного алгоритма генерации доменов) и legit (легитимный). Метки в столбце isDGA типа преобразуем в числовой формат: значения «dga» заменяются на 1, все остальные – на 0.

Производим группировку данных по классу subclass для доменов, относящихся и не относящихся к DGA, после чего сгруппированные доменные имена разбиваем на от-

дельные столбцы по уровням: первому и второму. Далее ко всем доменным именам в столбце `host` применяем функцию `split_domain` и на основе имен заполняем два новых столбца: `top_domain` и `second_domain`.

Выделяем признаки (X) и целевые переменные (Y): для обеспечения достаточного количества признаков для обучения моделей комбинируем значения столбцов `domain` и `host`. После чего разделяем данные на обучающую и те-

стовую выборки в пропорции, указанной в начале практической части исследования [10, 11].

Для преобразования текстовых данных в числовой формат создаем функцию на основе `TfidfVectorizer` из библиотеки `scikit-learn`. Задаем случайную базовую дату (было выбрано 19.07.2024) и генерируем временные метки в пределах одного дня (86 400 секунд) (см. рис. 2).

Target		Query		Time
0	1	wc5diss2ai2p7rk4qye13u6 wc5diss2ai2p7rk4qye13u...		2024-06-19 00:33:47
1	1	kb5fpy1iumnv71lmcazu2u41o kb5fpy1iumnv71lmca...		2024-06-19 23:28:29
2	1	qlahpubfqwu qlahpubfqwu.ru		2024-06-19 22:30:56
3	1	bp3zn8hy746110x3i5f11ps0l9 bp3zn8hy746110x3i5f...		2024-06-19 19:38:29
4	0	revenda5reais revenda5reais.com		2024-06-19 00:54:17
...	...		...	...
36994	1	r53953.tunnel.tuns.org.		2024-06-19 15:54:01
36995	1	r53954.tunnel.tuns.org.		2024-06-19 03:20:53
36996	1	r53955.tunnel.tuns.org.		2024-06-19 06:41:58
36997	1	r53956.tunnel.tuns.org.		2024-06-19 12:12:10
36998	1	r53957.tunnel.tuns.org.		2024-06-19 19:12:00

36999 rows x 3 columns

Рис. 2. Datasct validatingWithTime.csv

Создаем новый `DataFrame` `df_app` со столбцами `Target`, `Query` и `Time`, каждый из которых содержит следующие данные:

- `Target`: целевая переменная, указывающая на принадлежность домена к алгоритму DGA.
- `Query`: объединенное доменное имя.
- `Time`: сгенерированные временные метки в формате строки.

Полученные наборы данных позволяют перейти ко второму этапу исследования.

Для выбора ядра классификатора мы будем обучать три различные модели, после чего сравним их результаты:

- модель случайного леса (Random Forest)
- модель градиентного бустинга (Gradient Boosting)
- модель логистической регрессии (Logistic Regression)

Для каждой итерации (эпохи) обучения модели или слоя нейронной сети применяется метод кросс-валидации. Он заключается в случайном разбиении обучающей выборки доменных имен на десять блоков одинакового размера. После этого каждый блок поочередно рассматривается как тестовая выборка, а остальные девять – как обучающая. Далее таблица сопряженности рассчитывается как для каждого такого блока, так и средняя по всем 10 блокам.

Для преобразования доменных имен в последовательности чисел используем `Tokenizer` из `scikit-learn`. Токениза-

тор сначала обучается на столбце `domain` из исходного датасета, а потом преобразует доменные имена в числовые последовательности, которые затем приводятся к одинаковой длине с помощью `pad_sequences`.

Вызывается функция `create_dga_detection_model`, которая создает модель нейронной сети для бинарной классификации. Для сохранения лучшей модели на основе точности на валидационных данных настраивается колбэк `ModelCheckpoint`, куда передаются результаты работы модели.

Модели обучаются на обучающих выборках с заданным количеством эпох и размером пакета. Модель делает предсказания на тестовых данных. Результаты работы преобразуются в бинарный формат с использованием порога 0.5. Для визуализации результатов выводим точность, полноту и F1-меру для каждого бинарного класса, а также на основе `scikit-learn classification` выводим матрицу ошибок. Она показывает количество правильных и неправильных предсказаний для каждого класса. Визуализация помогает лучше понять, как каждая модель справляется с той или иной группой доменных имен.

Преобразуем доменные имена и хосты в числовые значения с помощью TF-IDF (Term Frequency-Inverse Document Frequency). Функция `vectorising` возвращает векторизованные данные для обучающей и тестовой выборок, а также сам результат преобразования для дальнейшего использования.

Создаем и обучаем модель `RandomForest`: здесь используется `scikit-learn RandomForestClassifier` со следующим набором

параметров: `nestimators` (количество деревьев), `maxdepth` (максимальная глубина дерева) и `minsamplesplit` (минимальное количество записей). Модель обучается на векторизованных данных `X_train_tfidf` и целевой переменной `y_train`.

Объединим некоторые subclasses доменных имен для обеспечения работы бинарного классификатора: доменные имена, относящиеся к категориям «alexa» и «legit», объединяем в категорию «alexa/legit», а все остальные приводим к значению «newgoz» – это наиболее популярный алгоритм генерации зараженных доменов.

Субклассы также преобразуются в числовые метки с помощью `scikit-learn LabelEncoder`, что позволяет модели взять фиксированное количество образцов из каждой группы и избежать перекоса в данных. Из сбалансированного набора данных `sampled_df` извлекаются признаки и целевая переменная. Здесь `X` содержит доменные имена, а `Y` содержит два столбца: `isDGA` и `subclass` из исходного датасета.

Создаем экземпляр модели `RandomForest` с параметром `nestimators=100` и визуализируем первый реальный результат предсказаний (см. рис. 3, рис. 4).

Сравним предсказанные subclasses доменных имен с их оригинальными значениями. Для этого создаем датафрейм со столбцами `original_subclass` и `predicted_subclass` и группируем записи их по оригинальным и предсказанным subclassам. Таким образом строки в датафрейме представляют оригинальные subclasses, а столбцы – предсказанные. Каждая ячейка содержит количество предсказаний для каждой комбинации строки и столбца (см. рис. 5).

Accuracy: 0.9346428571428571
Classification Report:

	precision	recall	f1-score	support
0	0.91	0.81	0.86	407
1	1.00	1.00	1.00	398
2	0.85	0.93	0.89	402
3	0.99	0.96	0.98	396
4	0.90	0.89	0.89	418
5	0.98	0.96	0.97	386
6	0.93	1.00	0.96	393
accuracy			0.93	2800
macro avg	0.94	0.94	0.94	2800
weighted avg	0.94	0.93	0.93	2800

Рис. 3. Результат работы модели случайного леса в разрезе точности и потерь

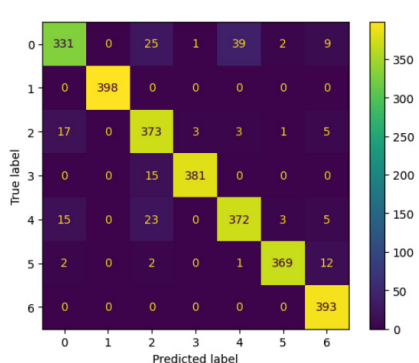


Рис. 4. Результат работы модели RandomForest (визуализация)

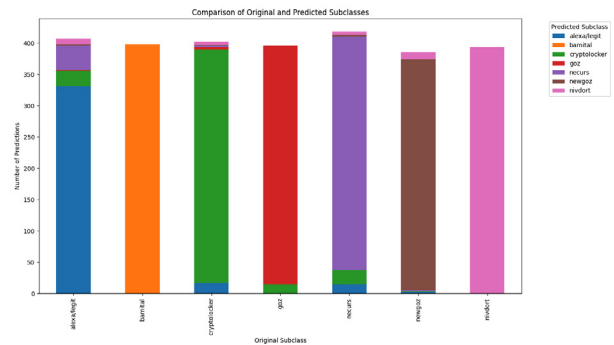


Рис. 5. Корреляция предсказанных и изначальных подклассов модели RandomForest с учетом их количества

Аналогичным образом создаем и обучаем модель `GradientBoosting`, которая последовательно строит множество слабых моделей, каждая из которых исправляет ошибки предыдущей. Аналогично `RandomForest` параметром `n_estimators=100` ограничиваем количество экземпляров. Визуализируем предсказания на тестовых данных (см. рис. 6, 7):

Accuracy: 0.9296428571428571
Classification Report:

	precision	recall	f1-score	support
0	0.90	0.82	0.86	407
1	1.00	1.00	1.00	398
2	0.85	0.92	0.89	402
3	0.99	0.96	0.97	396
4	0.89	0.89	0.89	418
5	0.99	0.92	0.96	386
6	0.91	0.99	0.95	393
accuracy			0.93	2800
macro avg	0.93	0.93	0.93	2800
weighted avg	0.93	0.93	0.93	2800

Рис. 6. Результат работы модели градиентного бустинга в разрезе точности и потерь

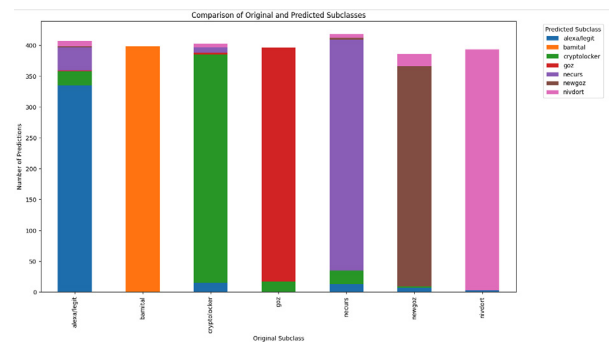


Рис. 7. Корреляция предсказанных и изначальных подклассов модели градиентного бустинга с учетом их количества

Финальный третий шаг в рамках данного этапа – обучение модели `LogisticRegression`. Параметром `max_iter=1000` указываем максимальное количество итераций для сходимости алгоритма. Результаты работы алгоритма представлены на рис. 8, 9, 10.

На этом этапе необходимо сделать несколько важных шагов, связанных с извлечением признаков, обучением моделей, их сохранением и анализом данных. Для этого выбираем случайную выборку из 50 000 строк датафрейма `df`, определяем признаки (`X1`) и целевую переменную (`Y1`) и оцениваем качество предсказаний (см. рис. 11, 12).

Accuracy: 0.9489285714285715
Classification Report:

	precision	recall	f1-score	support
0	0.94	0.85	0.89	407
1	0.99	1.00	1.00	398
2	0.92	0.90	0.91	402
3	0.98	1.00	0.99	396
4	0.91	0.91	0.91	418
5	0.97	0.99	0.98	386
6	0.92	1.00	0.96	393
accuracy			0.95	2800
macro avg	0.95	0.95	0.95	2800
weighted avg	0.95	0.95	0.95	2800

Рис. 8. Результат работы модели логистической регрессии в разрезе точности и потерь

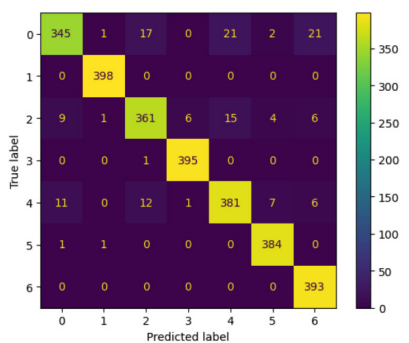


Рис. 9. Результат работы модели логистической регрессии в разрезе точности и потерь

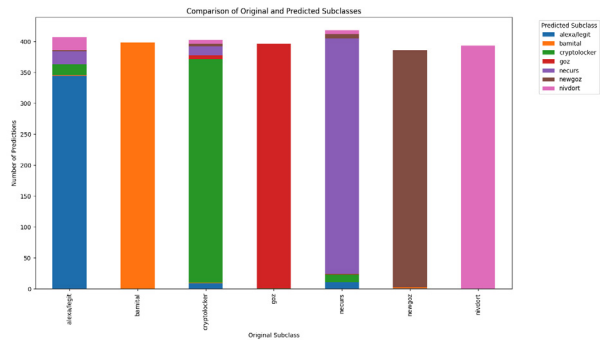


Рис. 10. Корреляция предсказанных и изначальных подклассов модели логистической регрессии с учетом их количества

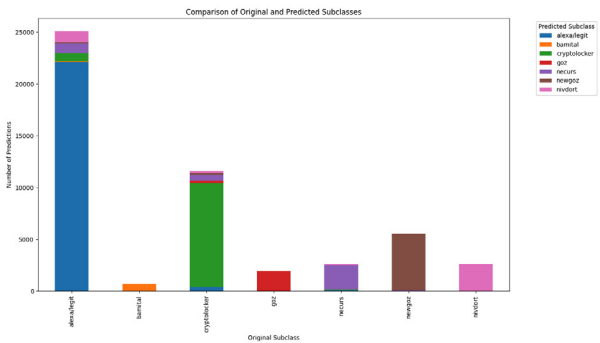


Рис. 11. Корреляция предсказанных и изначальных подклассов модели в случайной выборке

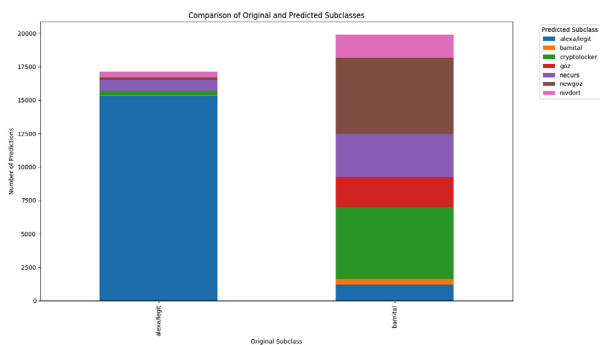


Рис. 12. Корреляция предсказанных и изначальных подклассов модели логистической регрессии после объединения подклассов

Приведем пример визуализации первых 10 строк для каждого из указанных подклассов (см. рис. 13).

```
Subclass: 4
13 rcdqeqmpxgdiuqombhuwcfu.ru
52 jrxrovxkbemgqutopblbbfe.ru
55 dixnzhelzukuwojfxuivlamadu.ru
84 ydhmbyhgypnkdpcymnnjxcalfm.ru
92 umficyzdvcdpzbaldmkjjvcn.ru
95 ztsclhitkplaemfiuucubozmr.ru
122 hmlzvbopfkgaztrsdexgjbageon.ru
126 pfvttkzllxnvngiobbeybayuolj.ru
134 lrbnqspjuuchnflvovrtolvpvy.ru
141 qrcrczfmxcqatfjzcypscxdxtts.ru
Name: host, dtype: object

Subclass: 7
3 zfd5szpi18i85wj9uy13l69rg.net
7 mzapiwbcbsbqiynlxalhprx8n.com
17 n3m4txqpb6a415iwrmdzgk6.com
37 wawft814u5se21tmptb51yy47q2.org
38 by86yc1p7f81d1tb5g7hie18t9k.org
42 jlkoy1mddlu81htl8cu13phhqf.org
44 fnbxw68p8e2l9qeq531wp37f2.com
54 1qkqxw21tgtfmw1bib3b9ccgacn.org
65 1m6kynyott35x196hh0av9i9wp.com
70 116b4hpzzekul1pzbwnit1f6u.com
Name: host, dtype: object
```

Рис. 13. Распределение доменов по подклассам

После преобразования текстовых значений в числовые обновляем файл векторизатора, на его основе извлекаем важные признаки из модели, подготавливаем новый набор данных для проверки и повторно проводим обучение моделей и оценку качества их работы на новых данных (см. рис. 14, 15).

Используем обученную модель LogisticRegression для предсказания меток классов на основе векторизованных данных. Результатом будет массив предсказанных классов (например, категории «dga» или «alexa/legit»). Подсчитываем количество предсказаний каждой метки моделями, для удобства анализа преобразовав полученное распределение в словарь (см. рис. 16, 17).

По итогам обучения и сравнения результатов нами была выбрана модель логистической регрессии как наиболее подходящая: точность по итогам обучения на обучающей и тестовой выборках составила 93%. Именно ее мы будем использовать в качестве ядра будущего классификатора.

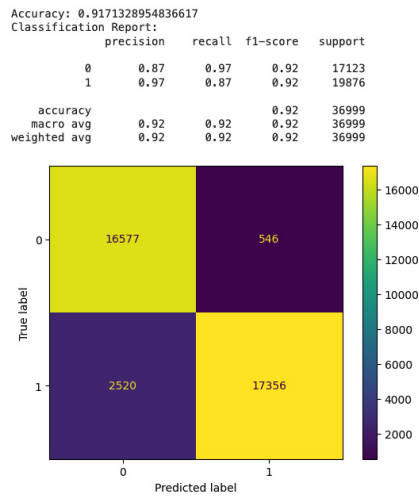


Рис. 14. Визуализация точности работы модели логистической регрессии при определении зараженных доменов на обучающей выборке

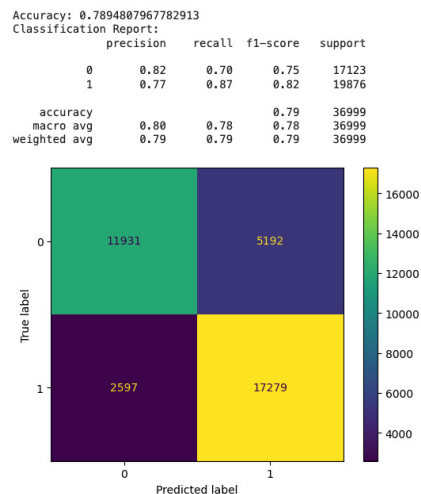


Рис. 15. Визуализация точности работы модели логистической регрессии при определении зараженных доменов на тестовой выборке данных

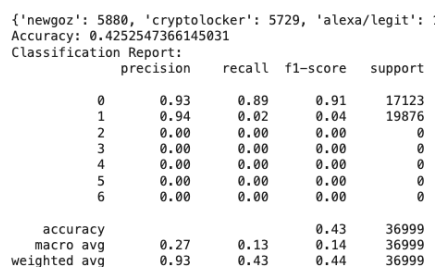


Рис. 16. Результат работы повторно обученной модели логистической регрессии в разрезе точности и потерь

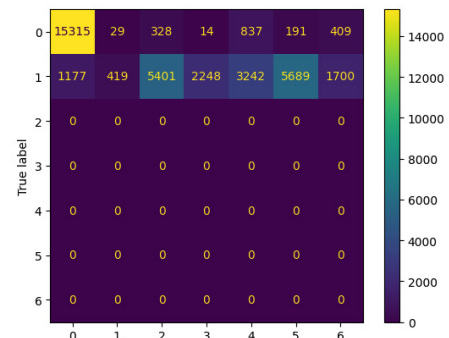


Рис. 17. Результат работы повторно обученной модели логистической регрессии (визуализация)

На финальном этапе исследования создадим и сравним две нейронные сети, впоследствии выберем ту, которая покажет наибольшую точность. Сети назовем последовательной и сигмоидальной.

Последовательная сеть содержит:

- два полносвязных слоя с активацией ReLU;
- слой Dropout для предотвращения переобучения;
- выходной слой с активацией softmax.

Визуализируем изменение точности и потерь на протяжении всех эпох обучения для анализа производительности модели нейронной сети и выявления возможного пере- или недообучения (см. рис. 18).

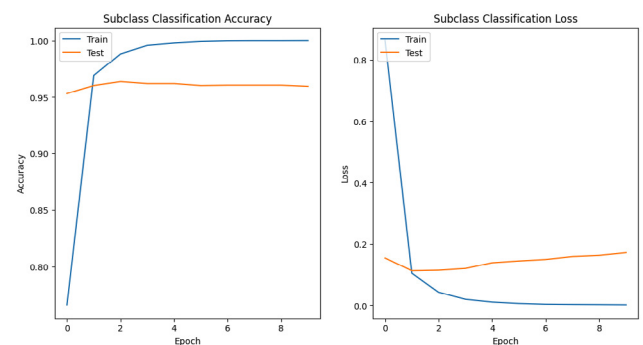


Рис. 18. Динамика точности и потерь модели последовательной нейронной сети на протяжении всех эпох обучения

Сигмоидальная нейронная сеть:

- входной слой, принимающий векторы признаков;
- два полносвязных слоя с активацией ReLU и слоями Dropout;
- выходной слой с сигмоидальной активацией.

Визуализация работы обеих сетей представлена на рис. 19.

Точность последовательной нейронной сети составила более 95,8%, исходя из чего можно сделать вывод о ее максимальной эффективности и об обеспечении безопасности сервера/ЦОД или иного объекта, в котором внедрен описанный алгоритм (рис. 20).

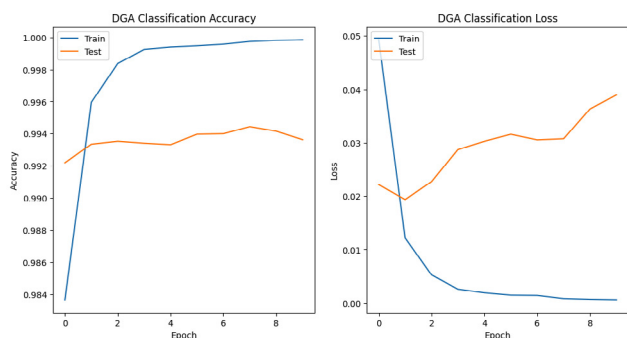


Рис. 19. Динамика точности и потерь модели сигмоидальной нейронной сети на протяжении всех эпох обучения

4. ЗАКЛЮЧЕНИЕ

Таким образом, в статье представлен алгоритм настройки бинарного классификатора, начиная с подготовки обучающих и тестовых данных и заканчивая преобразованием признаков в бинарные классы. В ходе тестирования было обнаружено взаимодействие объектов информационной системы с вредоносными доменами. Данные о вредоносных DNS-запросах позволяют администратору в реальном времени отслеживать попытки компрометации объекта информационной системы.

По результатам проведенных исследований был разработан и программно реализован двухэтапный алгоритм обнаружения вредоносных DGA-доменов: на базе сравнения моделей Random Forest, Gradient Boosting и Logistic Regression встроенными методами библиотеки scikit-learn

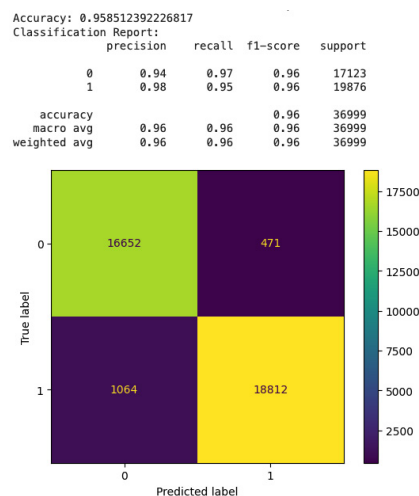


Рис. 20. Финальный результат последовательной нейронной сети

с помощью методов машинного обучения происходит классификация доменных имен и последующее принятие решения о характере DNS-запросов. Ядро программного решения применимо для большинства операционных систем и разновидностей оборудования.

В результате проведенного анализа можно сделать вывод об эффективности бинарной классификации в задачах выявления DGA-доменов и ее успешной интеграции в ИС для защиты пользователей и сетевой инфраструктуры.

К статье прилагается QR-код со ссылкой на материалы статьи (кодированную базу, исходный датасет, визуализацию работы моделей на всех этапах):



Список использованных источников и литературы

1. Галияхметов Д.Г. Сравнение алгоритмов классификации применительно к задаче обнаружения вредоносных доменных имен // Математические методы в технике и технологиях-MMTT. 2019. Т. 12. С. 190–194.
2. Berman D.S., et al. A survey of deep learning methods for cyber security // Information. 2019. Т. 10, № 4. С. 122.
3. Бубнов Я.В., Иванов Н.Н. Обнаружение DGA доменов и предотвращение botnet средствами Q-обучения для POMDP // Доклады Белорусского государственного университета информатики и радиоэлектроники. 2021. Т. 19, № 2. С. 91–99.
4. Antonakakis M., Perdisci R. From throw-away traffic to bots: detecting the rise of DGA-based malware // In Proceedings of the 21st USENIX Security Symposium. 2012, pp. 491–506.
5. Binkley J.R., Singh S. An algorithm for anomaly-based botnet detection // SRUTI. 2006. № 6. pp. 43–48.
6. Li Y., Xiong K. Machine Learning Framework for Domain Generation Algorithm-Based Malware Detection // IEEE Access. 2019. С. 32765–32782.
7. Alazab M., Tang M. Deep Learning Applications for Cyber Security. Springer Nature Switzerland, 2019. 246 с.
8. AsSadhan B., Moura J.M.F., Lapsley D., et al. Detecting botnets using command and control traffic // In 2009 8th IEEE International Symposium on Network Computing and Applications. IEEE, 2009. С. 156–162.

9. Malwarebytes Labs 2019 State of Malware Report. URL: <https://resources.malwarebytes.com/files/2019/01/Malwarebytes-Labs-2019-State-of-Malware-Report-2.pdf> (дата обращения: 15.05.2025).
10. Wang Z., Jia Z., Zhang B. A detection scheme for DGA domain names, based on SVM // In 2018 International Conference on Mathematics, Modelling, Simulation and Algorithms (MMSA 2018). 2018. С. 257–263.
11. Why Machine Learning Models Degrade in Production [Электронный ресурс]. URL: <https://towardsdatascience.com/why-machine-learning-models-degrade-in-production-d0f2108e9214> (дата обращения: 15.05.2025).

APPLICATION OF MACHINE LEARNING METHODS TO COUNTER DGA THREATS IN THE ENTERPRISE INFORMATION SECURITY QUALITY MANAGEMENT SYSTEM

Korolev I.A., MSc, Financial University under the Government of the Russian Federation, Moscow

Kobelev E.A., MSc, Financial University under the Government of the Russian Federation, Moscow

Bulgakova E.V., Associate Professor, Ph.D., Associate Professor, Moscow University of the Ministry of Internal Affairs of Russia named after V.Ya. Kikot, Moscow

Kubankov A.N., Professor, Doctor of Military Sciences, Chief Researcher, Russian Standardization Institute, Moscow

This article defines the concept of DGA technology, analyzes existing solutions for recognizing its application, and forms a set of machine learning models suitable for identifying infected domains. Based on the model comparison and training, a neural network is developed for automatic data classification.

Methods used: This study focuses on the development and implementation of an algorithm for detecting DGA domains using machine learning models. It describes the binary classifier setup, including feature transformation, model training, and performance interpretation.

Results of the study: Three models – Random Forest, Gradient Boosting, and Logistic Regression – were compared. Gradient Boosting showed the best performance (ROC AUC = 0.998).

Practical value: The proposed solution is suitable for integration into DNS monitoring systems to enhance information security. The developed algorithm can be applied in real time and integrated into information systems to protect network infrastructure.

Keywords: DGA, domain, host, botnet, machine learning, neural network, binary classification.

For citation: Korolev I.A., Kobelev E.A., Bulgakova E.V., Kubankov A.N. Application of Machine Learning Methods to Counter DGA Threats in the Enterprise Information Security Quality Management System. Information and Economic aspects of standardization and technical regulation. 2025; 4(85): 46–55. (In Russ.).

References

1. Galiakhmetov D.G. Sravnenie algorytmov klassifikatsii primenitel'no k zadache obnaruzheniya vredonosnykh domennykh imyon. *Matematicheskie metody v tekhnike i tekhnologiyah-MMTT*. 2019; 12: 190–194. (In Russ.).
2. Berman D.S., et al. A survey of deep learning methods for cyber security. *Information*. 2019; 10(4): 122.
3. Bubnov Y.V., Ivanov N.N. Obnaruzheniye DGA domenov i predotvrascheniye botnet sredstvami Q-obucheniya dlya POMDP. *Doklady Belorusskogo gosudarstvennogo universiteta informatiki i radioelektroniki*. 2021; 19(2): 91–99. (In Russ.).
4. Antonakakis M., Perdisci R., et al. From {Throw-Away} traffic to bots: Detecting the rise of {DGA-Based} malware. In *21st USENIX Security Symposium*. 2012, pp. 491–506.
5. Binkley J.R., Singh S. An algorithm for anomaly-based botnet detection. *SRUTI*. 2006; 6: 43–48.
6. Li Y., Xiong K. Machine Learning Framework for Domain Generation Algorithm-Based Malware Detection. *IEEE Access*. 2019, pp. 32765–32782.
7. Alazab M., Tang M. eds. *Deep Learning Applications for Cyber Security*. Springer, 2019, 246 p.
8. AsSadhan B., Moura J.M.F., Lapsley D., et al. Detecting botnets using command and control traffic // In *2009 8th IEEE International Symposium on Network Computing and Applications*. IEEE, 2009, pp. 156–162.
9. Malwarebytes Labs 2019 State of Malware Report. URL: <https://resources.malwarebytes.com/files/2019/01/Malwarebytes-Labs-2019-State-of-Malware-Report-2.pdf> (Accessed 15 may 2025).
10. Wang Z., Jia Z., Zhang B. A detection scheme for DGA domain names, based on SVM // In *2018 International Conference on Mathematics, Modelling, Simulation and Algorithms (MMSA 2018)*. 2018, pp. 257–263.
11. Why Machine Learning Models Degrade in Production. URL: <https://towardsdatascience.com/why-machine-learning-models-degrade-in-production-d0f2108e9214> (Accessed 15 may 2025).