**Обзор методов машинного обучения и анализа данных для обнаружения вторжений в кибербезопасности**

***Аннотация*. В этой обзорной статье описывается специализированный обзор литературы по методам машинного обучения (ML) и интеллектуального анализа данных (DM) для кибер-аналитики для обеспечения обнаружения вторжений. Предоставляются краткие описания каждого метода ML / DM. Основываясь на количестве цитат или актуальности возникающего метода, работы, представляющие каждый метод, были идентифицированы, прочитаны и обобщены. Поскольку данные так важны в подходах ML / DM, описаны некоторые известные наборы кибер-данных, используемые в ML / DM. Рассматривается сложность алгоритмов ML / DM, представлено обсуждение проблем использования ML / DM для кибербезопасности и приводятся некоторые рекомендации о том, когда использовать данный метод.**

***Индексные термины* - кибер-аналитика, интеллектуальный анализ данных, машинное обучение.**

Данная работа представляет результаты обзора литературы по методам машинного обучения и анализа данных в приложениях для кибербезопасности. Эти методы представлены, а так же несколько применений каждого метода к задачам обнаружения вторжений. Обсуждается сложность различных алгоримтов, а работа предоставляет ряд сравнительных критериев этих методов и ряд рекомендаций +по лучшим методоам, чтобы использовать их в зависимости от характеристик решаемой кибер-проблемы.

Кибербезопасность - это ряд технологий и процессов, разработанных для защиты компьютеров, сетей программ и данных от атаки, несанкционированного доступа, изменения или разрушения. Системы кибербезопасности состоят из систем сетевой безопасности и систем компьютерной (доменной) безопасности. Каждая из них имеет, по крайней мере, файервол, антивирусное программное обеспечение и систему обнаружения вторжений. СОВ помогает, обнаруживать, определять и распознавать несанкционированное использование, дублирование, внесение изменений и уничтожение информационных систем [1]. Нарушения безопасности включают в себя внешние вторжения (атаки извне организации) и внутренние вторжения (атаки изнутри организации).

Существует три основных типа кибераналитики в поддержку СОВ: **неправомерное использование** (иногда также называют сигнатурные), основанные на аномалиях и гибридные. **Техники, основанные на неправомерном использовании** разрабатываются для обнаружения известных атак, используя сигнатуры этих атак. Они эффективны для обнаружения известных типов атак без генерации подавляющего количества ложных срабатываний. Они требуют частых ручных обновлений базы данных, содержащей правила и сигнатуры. Техники, основанные на неправомерном использовании, не могут обнаруживать новые атаки (уязвимости нулевого дня).

Методы, основанные на аномалиях, моделируют нормальную сеть и поведение системы и распознают аномалии как отклонения от нормального поведения. Они привлекательны из-за их способности обнаруживать атаки нулевого дня. Другим преимуществом является то, что профили нормальной активности настраиваются для каждой системы, приложения или сети, **тем самым затрудняя для злоумышленников получение представения о том, что они могут выполнять незамеченными.** Дополнительно, данные, на которых методы, основанные на аномалиях, срабатывают (новые атаки) могут быть использованы для определения сигнатур для детекторов неправомерного использования. **Основным недостатком методов, основанных на аномалиях, является вероятность высокой частоты ложных срабатываний, потому что ранее неизвестные (но законные) поведения системы могут быть классифицированы как аномалии.**

Гибридные методы совмещают в себе обнаружение неправомерного использования и аномалии. Они служат для повышения уровня обнаружения известных вторжений и снижения вероятности ложных срабатываний для неизвестных атак. Углубленный обзор литературы не обнаружил многих чистых методов обнаружения аномалии; большинство методов были действительно гибридными. Поэтому в описаниях методов машинного обучения и анализа данных методы обнаружения аномалий и гибридные методы описаны вместе.

Другой раздел СОВ основан на том, где системы ищут модель **поведения, похожую на вторжение**: сеть или домен. СОВ, **основанные на поиске в сети**, обнаруживают вторжения, **исследуя трафик, проходящий через сетевые устройства.** СОВ, **основанные на поиске в домене, контролируют процесс и действия файла, связанные с программной средой, которая в свою очередь связана с конкретным доменом.**

Эта обзорная статься фокусируется на методах машинного обучения и анализе данных для кибербезопасности, **ориентирована на методы машинного обучения и анализа данных и их описаниях. Было уже опубликовано множество работ, описывающих эти методы, а также включало некоторые обзоры.** В отличие от предыдущих обзоров, основное внимание в нашей статье уделяется публикациям, которые соответствуют определенным критериям. Запросы Google Scholar выполнялись с использованием «машинного обучения» и кибер, а также с использованием «анализа данных» и кибер. Особый внимание было уделено высоко цитируемым статься, поскольку они описывали популярные методы. **Однако, также было признано, что эта задача может не учитывать важные появившиеся методы, поэтому некоторые из этих работ также были выбраны. В целом, статьи были отобраны таким образом, чтобы каждая из перечисленных категорий машинного обучения и анализа данных имели по меньшей мере одну, а лучше несколько представленных работ.**

Эта статься предназначена для читателей, которые хотят начать исследование в области машинного обучения и анализа данных для обнаружения кибервторжений. **Таким образом, значительный акцент делается на подробное описание методов машинного обучения и анализа данных, и представлены ссылки на коснтруктивные работы для каждого метода машинного обучения и анализа данных. Несколько примеров представлены на тему того, как методы используются в кибербезопасности.**

Эта работа не описывает все различные методы обнаружения сетевых аномалий, как это делает Bhuyan и другие [2]; вместо этого она концентрируется только на методах машинного обучения и анализа данных. Однако, в дополнение к обнаружению аномалий, изображены сигнатурные и гибирдные методы. Описания методов в данном обзоре являются более углубленным чем в [2].

Nguyen и другие [3] описывают методы машинного обучения для классификации Интернет трафика. Методы, описанные в нем, зависят не от известных номеров портов, а от статистических характеристик трафика. Их обзор охватывает работы, опубликованные только с 2004 по 2007, когда наш обзор включает **более новые работы**. В отличие от работы Nguyen и других [3], эта статься представляет методы, которые работают на любом типе данных, **а не только на потоках протокола Интернет**.

Teodoro и другие [4] фокусируются на методах обнаружения сетевых аномалий. Авторы представляют статистические, основанные на знаниях и подходах в машинном обучении, но их исследование не представляет полный набор современных методов машинного обучения. В отличие от них, эта работа описывает не только обнаружение аномалий, но также и сигнатурные методы. Наша работа также включает в себя методы для распознавания типов атак (неправомерное использование) и для обнаружения атак (вторжение). Наконец, в нашей работе представлен полный и последний список методов машинного обучения и анализа данных, которые применяются для кибербезопасности.

Sperotto и другие [5] фокусируются на данных сетевого потока и указывают на то, что обработка пакетов может быть невозможна на скорости потоковой передачи из-за количества трафика. Они описывают широкий набор методов для обнаружения аномального трафика (возможная атака) и сигнатур. Однако, в отличие от нашей работы, они не включают объяснений технический деталей отдельных методов.

Wu и его коллеги [6] уделяют основное внимание методам вычислительного интеллекта и **их применениям к обнаружению вторжений**. Такие методы, как искусственные нейронные сети (ИНС), нечеткие системы, эволюционные вычисления, искусственные иммунные системы и роевой интеллект, описаны очень подробно. Поскольку описываются только методы вычислительного интеллекта, основные методы машинного обучения и анализа данных, такие как кластеризация, деревья решений и анализ правил (**которые не рассматриваются в данной работе**) не включены.

В данной статье основное внимание уделяется обнаружения кибервторжений, применительно к проводным сетям. В проводной сети, **злоумышленник** должен пройти через несколько уровней защиты в брэндмауэрах и операционных системах или получить физический доступ к сети. Однако, беспроводная сеть может быть нацелена на любой узел, поэтому она, естественно, более уязвима к вредоносным атакам, чем проводная сеть. Методы машинного обучения и анализа данных, описанные в этой работе, полностью применимы к проблема обнаружения вторжений и неправомерного использования как в проводных, так и беспроводных сетях. Читатель, **которому необхдима только защита беспроводной сети, может обратиться к таким работам как Zhang и другие [7], в которой основное внимание уделяется динамическому изменению топологии сети, алгоритмам маршрутизации, децентрализованному управлению и т.д.**

Остальная часть этой работы организована следующим образом: секция II фокусируется на основным шагам в машинном обучении и анализе данных. В секции III обсуждаются наборы данных кибербезопасности, которые используются в машинном обучении и анализе данных. В секции IV описываются отдельные методы и связанные с ними работы для машинного обучения и анализа данных в кибербезопасности. В секции V обсуждается вычислительная сложность различных методов. В секции VI описываются замечания и рекомендации. Наконец, в секции VII представлены заключения.

**Основные шаги в машинном обучении и анализе данных**

**Существует много путаницы в терминах машинное обучение, анализ данных и поиск знаний в базах данных**. ПЗБД это полный процесс, который ориенитирован на извлечением полезной, ранее неизвестной информации (то есть знания) из данных [8]. Анализ данных это конкретный шаг в этом процессе — **применение** конкретных алгоритмов для извлечения шаблонов из данных. Дополнительные шаги в процессе ПЗБД (подготовка данных, выбор данных, очистка данных, включение соответствующих предварительных знаний и правильная интерпретация результатов анализа данных) гарантируют, что полезные знания будут извлечены из доступных данных. Однако, существуют множество публикаций (например, межотраслевой стандарт для анализа данных [9]) и участников этой отрасли, которые называют весь процесс ПЗБД анализом данных. В этой работе, следуя Fayyad и другие [8], анализ данных используется для описания конкретного этапа в ПЗБД, который касается применения конкретных алгоритмов для извлечения шаблонов из данных.

Существует значительное совпадение между машинным обучением и анализом данных. Эти два термина обычно путают, потому что они часто используют одни и те же методы и поэтому значительно совпадают. Пионер машинного обучения, Arthur Samuel, определил машинное обучение как «область исследования, которая дает компьютерам возможность обучаться, не будучи явно запрограммированными». Машинное обучение фокусируется на классификации и прогнозировании, основываясь на известных свойствах, извлеченных из обучающих данных. Алгоритмы машинного обучения нуждаются в цели (постановка задачи) из области (например, зависимая переменная для предсказания). Анализ данных уделяет основное внимание обнаружению ранее неизвестных свойств в данных. Он не нуждается в конкретной цели из области, но вместо этого он напавлен на поиске новых и интересных знаний.

Машинное обучение можно рассматривать как старшего брата анализа данных. Термин анализ данных был введен в конце 1980-х годов ( первая ПЗБД конференция состоялась в 1989 году), тогда как термин машинное обучение используется с 1960-х годов. В настоящее время, младший брат (то есть использование термина анализ данных) **более популярен, чем уже существующий**, **что может быть причиной того, что некоторые исследователи на самом деле называют свою работу анализ данных, а не машинное обучение.** Причиной этого могло быть то, что когда в Google Scholar выполнялись запросы «машинное обучение» И кибер и «анализ данных» И кибер, первый запрос возвращал 21300 результатов, а второй — 40800 результатов. Методы, использовавшиеся в работах, полученных по первому запросу, существенно не отличались от методов, полученных вторым запросом. Поэтому, поскольку эта работа концентрируется на методах, мы будем называть их методами машинного обучения и анализа данных.

Подход машинного обучения обычно состоит из двух фаз: обучение и тестирование. Часто, выполняются следующие этапы:

* **Идентификация** атрибутов класса (признаки) и классов в обучающих данных.
* **Идентификация** подмножества атрибутов, необходимых для классификации (то есть уменьшение размерности).
* Обучение модели, используя обучающие данные.
* Использование обученной модели для классификации неизвестных данных.

В случае обнаружения сигнатур, в обучающей фазе каждая сигнатурный класс обучают, используя подходящие примеры из обучающего набора. На этапе тестирования, новые данные пропускаются через модель и образец классифицируют относительно того, принадлежит ли он к одному из сигнатурных классов. Если экземпляр не принадлежит ни одному из сигнатурных классов, он классифицируется как нормальный.

В случае обнаружения аномалий, шаблон нормального трафика определяют на этапе обучения. В фазе тестирования, обученную модель применяют к новым данным, и каждый экземпляр в обучающем наборе классифицируют либо как нормальный, либо аномальный.

В действительности, для большинства методов машинного обучения должно существовать три этапа, а не два: обучение, проверка и тестирование. Машинное обучение и анализ данных часто имеют такие параметры как число слоев и узлов для ИНС. После того, как обучение завершится, обычно имеется несколько доступных моделей (то есть ИНС). Чтобы решить, какая из них **будет использоваться и иметь хорошую оценку ошибки, которую она получит на тестовом наборе данных, должен быть третий отдельный набор данных для проверки.** Модель, которая наилучшим образом использует проверочные данные, должна быть используемой моделью **и не должна быть точно настроена в зависимости от ее точности в наборе тестовых данных.** В противном случае, **представленная точность является оптимистичной и может не отражать точность, которая будет получена на другом аналогичном тестовом наборе, но немного отличающемся от существующего тестового набора.**

Существует три основных типа подходов машинного обучения и анализа данных: обучение без учителя, **полуконтролируемое обучение,** обучение с учителем. В проблемах обучения без учителя, основной задачей является нахождение шаблонов, структур или знаний в немаркированных данных. Когда часть данных помечается во время сбора данных или специалистами, проблему называют частичным обучением. Дополнение маркированных данных значительно помогает решить проблему. Если данные полностью помечены, проблема называется обучением с учителем и, в целом, задачей является поиск функции или модели, которая объясняет эти данные. Такие подходы как подгонка кривой или методы машинного обучения используются для моделирования данных для основной проблемы. Метка, как правило, представляет собой **переменную бизнеса или задачи**, которая, по мнению экспертов, имеет отношения к собранным данным.

Как только модель классификации будет разработана с использование обучающих и проверочных данных, модель может быть сохранена, чтобы ее можно было использовать позднее или на другой системе. Язык разметки прогнозного моделирования (**ЯРПМ**) разработан и предложен Data Mining Group, чтобы облнгчить **интеллектуальное моделирование** [10]. Он основан на XML и в данный момент поддерживает логистическую регрессию и классификаторы нейронных сетей с прямыми связями. Последняя версия (4.2) поддерживает **Наивный Байесовский классификатор, k-ближайших соседей и метод опорных векторов.** Модель поддерживает несколько распространенных метаданных анализа данных, таких как словарь данных (например, дискретный, логический, численный), нормализация, имя модели, атрибуты модели, схема анализа, обработка выбросов и вывод. Некоторые популярные платформы для анализа данных, такие как Weka [11], R [12] и RapidMiner [13] поддерживают модели ЯРПМ.

Модель CRISP-DM [9] иллюстрирует (см рис. 1) часто используемые этапы и парадигмы экспертами анализа данных для решения проблем.

Модель составлена из следующих шести этапов:

* *Понимание бизнеса:* Определение проблемы анализа данных в соответствии с требованиями проекта.
* *Понимание данных:* Сбор и **изучение** данных.
* *Подготовка данных:* Все аспекты подготовки данных для **получения** окончательного набора данных.
* *Моделирование:* Применение методом анализа данных и машинного обучения и оптимизация параметров в соответствии с лучшей моделью.
* *Оценка:* **Оценка метода с подходящими метриками для проверки того, что бизнес-цели достигнуты.**
* *Развертывание:* **Варьируется от подтверждения отчета до полной реализации сбора данных и моделирования структуры. Обычно, аналитик данных занимается этапами до развертывания, в то время как клиент выполняет само развертывание.**

Существует несколько метрик классификации для методов машинного обучения и анализа данных. Некоторые метрики называют двумя или даже тремя различными именами. В секции IV, представлены работы с именами метрик, используемых авторами соответствующих статей. Чтобы легче понять этот раздел, метрики с их различными именами описаны далее. Для проблемы бинарного классификатора, метрики вычисляются из матрицы неточностей (см. Таблица 1).

Метрики часто используемые для задач бинарной классификации (обучение с учителем):

* Точность или **Proportion Correct: (TP +TN) / (TP + TN + FP + FN)**. Когда классы сбалансированы, это хорошая мера; однако, когда классы не сбалансированы (то есть 97% **элементов** принадлежат к классу Х и 3% к классу Y: если все **элементы** классифицируются как Х, точность будет равна 97%, но все элементы из класса Y будут ошибочно классифицированы), такая метрика не очень полезна.
* **Положительное прогнозируемое значение или Прогнозирование**: **TP / (TP + FP)**. Соотношение **пунктов**, правильно классифицированных как Х, ко всем **пунктам**, классифицированным как Х.
* Чувствительность или Отзыв или **True Positive Rate** или Вероятность обнаружения (**PD**) или **Скорость обнаружения (Detection Rate)**: **TP / (TP + FN)**. **Соотношение элементов (Ratio of items),** правильно классифицированных как Х, ко всем элементам, которые принадлежат классу Х.
* **Отрицательное прогнозируемое значение (Negative Predictive Value) (NPV): TN / (TN + FN). Соотношение элементов (Ratio of items)**, правильно классифицированных как отрицательные (не Х), ко всем элементам, классифицированных как не Х.
* **Specifity** или **TN Rate: TN / (TN + FP).** Соотношение элементов, правильно классифицированных как отрицательные (не Х), ко всем элементам, которые принадлежат не классу Х.
* **FAR or FP Rate or Fall-out: FP / (TN + FP). FAR = 1-Specifity.** Соотношение элементов, неправильно классифицированных как положительные (Х), ко всем элементам, которые принадлежат не классу Х.

**In classification problems, there is a trade-off between Sensitivity and FAR (1-Specificity). This trade-off is illustrated by a Receiver Operating Characteristic (ROC) curve. ROC has FAR on the x-axis and Sensitivity on the y-axis. As the threshold for classification is changed, a different point on the ROC is chosen with different FAR and different Sensitivity. A higher FAR results in a higher Sensitivity and a lower FAR in a lower Sensitivity. The point on the ROC that provides a better classi- fication is application dependent. Often, FAR cannot be higher than a certain number, and this is how the final classifier is chosen.**

Для многоклассовой задачи (классификация **по более чем двум классам (into more than two classes)**) обычноиспользуются следующие метрики:

* **Общая точность (Overall accuracy)**: Экземпляры классифицированы правильно, все экземпляры.
* **Уровень обнаружения класса (Class detection rate)**: Экземпляры из полученного класса классифицированы правильно, все экземпляры из полученного класса.
* **Уровень класса FAR** **класса** **FR (Class FAR or class FP rate)**: Экземпляры из полученного класса классифицированы неправильно, все экземпляры не из данного файла.

Можно также вычислить **PPV и NPV** для каждого класса, но в работах, рассмотренных и описанных в секции IV, эти метрики не использовались.

Существует два типа метрик для методов обучения без учителя: внутренняя и внешняя. Внутренние метрики используются на данных, которые кластеризованы, и метки класса (потому что они неизвестны для алгоритма кластеризации) не используются для вычисления этих метрик. Такие метрики как межкластреное расстояние (расстояние между двумя различными кластерами, могут быть между их центроидами (центры кластеров)), внутреннее расстояние (расстояние между членами одного и того же кластера, может быть среднее расстояние между самыми дальними членами) и индекс Данна (определяет **плотность (dense)** и хорошо разделенные кластеры) часто используются.

Внешние метрики работают с набором данных, для которого метки класса известны. Используемые метрики напоминают **метрики для обучения с учителем (the supervised learning metrics).** Некоторые из работ, описанные в секции IV, используют **Class Detection Rate и the Class FAR.** Это означает, что хотя метод был разработан в стиле обучения без учителя, для тестовых данных были доступны метки, поэтому было возможно вычислить метрики классификации.

**Наборы данных по кибер-безопасности для машинного обучения и анализа данных**

**(Cyber-Security data sets for ML and DM)**

Для подходов машинного обучения и анализа данных, данные имеют большое значение. Потому что эти методы обучаются из доступных данных, необходимо иметь представление о данных, которые они используют, чтобы понять, как разные авторы применяют различные алгоритмы машинного обучения и анализа данных. Этот раздел подробно описывает различные типы данных, используемые подходами машинного обучения и анализа данных — захват пакетов (**packet capture (pcap))**, **NetFlow** и другие сетевые данные. Поэтому, в разделе IV, в котором подробно описываются методы, цитируется только то, используются ли методы pcap, NetFlow или другие сетевые данные и не описывает эти данные подробно. Следующие подразделы охватывают детали низкоуровневого набора данных.

А. **Данные на уровне пакетов (Packet-Level Data)**

Существует 144 IP-адреса, перечисленных Инженерным советом интернета (**Internet Engineering Task Force (IETF)**), которые включают такие широко используемые протоколы как **Transmission Control Protocol (TCP, протокол управления передачей)**, **User Datagram Protocol (UDP, протокол пользовательских датаграмм), Internet Control Message Protocol (ICMP, протокол межсетевых управляющих сообщений), Internet Gateway Management Protocol (IGMP, протокол управления групповой передачей данных)** и т.п. Пользовательские программы, запускающие эти протоколы, генерируют пакетный сетевой трафик Интернета. Сетевые пакеты, принимаемые и передаваемые на физическом интерфейсе (например, Ethernet-порт) компьютера, могут быть захвачены с помощью специального интерфейса прикладного программирования (**API**), называемого pcap. Libpcap и WinPCap (версии для Unix и Windows, соответственно) – **это библиотеки для перехвата пакетов для многих сетевых инструментов (are the front-end packet capture software libraries for many network tools)**, **включающие анализаторы протоколов, пакетные снифферы, сетевые мониторы, сетевые СОВ и генераторы трафика (including protocol analyzers, packet sniffers, network monitors, network IDSs, and traffic generators)**. Несколько популярных программ, которые используют данные pcap: tcpdump [14], Wireshark [15], Snort [16] и Nmap [17].

На сетевом физическом уровне, Ethernet-фрейм состоит из заголовка Ethernet (то есть адреса управления доступом к среде передачи (**Media Access Control, МАС**)) и до 1500 байтов (максимальная единица передачи ((**Maximum Transmission Unit, MTU**)) полезной нагрузки. Эта полезная нагрузка содержит IP-пакет, который состоит из заголовка IP (то есть транспортного уровня) и полезной нагрузки IP. Полезная нагрузка IP может содержать данные или другие инкапсулированные протоколы высокого уровня, такие как сетевая файловая система (**Network File System, NFS**), **Server Message Block (SMB), Протокол передачи гипертекста (Hypertext Transfer Protocol**, **HTTP), BitTorrent,** протокол почтового отделения (**Post Office Protocol, POP**) версии 3, сетевая базовая система ввода/вывода (**Network Basic Input/Output System, NetBIOS**), telnet и простой протокол передачи данных (**Trivial File Transfer Protocol, TFTP**).

Поскольку весь пакет захвачен интерфейсов pcap, признаки данных различаются в отношении протокола, который передает пакет. В таблице 2 перечислены подмножества захваченных признаков для TCP, UDP и ICMP. IP-адреса находятся в заголовке IP, которые обрабатываются на сетевом уровне.

B. **Данные NetFlow**

Первоначально, NetFlow ,был представлен как маршрутизатор от Cisco. Маршрутизатор или коммутатор обладает способностью собирать IP-трафик, по мере того, как он входит и выходит из интерфейса. NetFlow от Cisco версии 5 определяет сетевой поток как однонаправленную последовательность пакетов, которые одинаково разделены на семь атрибутов: входной интерфейс, IP-адресс источника, IP-адрес получателя, протокол IP, порт источника, порт получателя и тип обслуживания IP. Логическая архитектура NetFlow состоит из трех компонентов: **сенсор NetFlow, коллектор NetFlow и консоль анализатора (a NetFlow Exporter, a NetFlow Collector, and an Analysis Console)**. На данные момент, существует 10 версий NetFlow. Версии с 1 по 8 аналогичны, но начиная с 9 версии, NetFlow значительно отличается. Для версий с 1 по 8, набор признаков в таблице 3 представлен минимальный набор переменных данных NetFlow для однонаправленной последовательности пакетов (то есть потока).

Данные NetFlow включают в себя сжатую и предварительно обработанную версию **настоящих (actual)** сетевых пакетов. Статич\стические данные это обнаруженные признаки, основанные на, установленного набора настроек на устройстве и,**The statistics are derived features and, based on certain parameters such as duration of window, number of packets, etc., set the NetFlow settings on the device.**

C. **Открытые наборы данных (Public Data Sets)**

Наборы данных DARPA 1998 и DARPA 1999 [18], [19] широко используются в экспериментах и часто цитируются в публикациях. Набор DARPA 1998 был создан **группой киберсистем и технологий лаборатории Линкольна Массачусетского технологического института** **(the Cyber Systems and Technology Group of the Massachusetts Institute of Technology Lincoln Laboratory, MIT/LL)**. Была построена симуляционная сеть и данные были скомпилированы на основе сетевых данных TCP/IP, данных журнала Solaris Basic Security Module (Solaris BSM) и дампов файловой системы Solaris для пользователя и root-пользователя. Фактически собранный набор данных составлен из данных сети и операционной системы (ОС). Данные были собраны в течение 9 недель, причем первые 7 были назначены в качестве набора для обучения, а последние 2 - в качестве набора тестов. Симуляции атак были организованы **во время недель обучения и тестирования.**

Аналогично, набор данных DARPA 1999 был собран в общей сложности за 5 недель, причем первые 3 были назначены в качестве набора для обучения, а последние 2 - в качестве набора для тестирования. Этот набор данных имел значительно больше типов атак, чем набор данных DARPA 1998. В обеих коллекциях наборы данных обрабатывались и **выбирались** для использования в экспериментах. Дампы и журналы TCP были объединены в один поток со многими столбцами.

Одним из наиболее широко используемых наборов данных является набор данных KDD 1999 [20], который был создан для **задачи** KDD Cup в 1999 году. Набор данных **основан** на данных TCP/IP DARPA 1998 и имеет основные признаки, захваченные **с помощью** pcap. Дополнительные признаки были получены путем анализа данных с помощью временных и последовательных окон. Набор данных имеет три компонента: **базовые, контентные и трафик-признаки**, что составляет в общей сложности 41 атрибут. Набор данных KDD 1999 аналогичен данным NetFlow, но имеет больше полученных и подробных признаков, **поскольку атаки были смоделированы**. Полный список можно найти в таблице IV.

Набор данных KDD 1999 (примерно с 4 миллионами записей о нормальном и атакующем трафике) был всесторонне проанализирован Tavallaee и другими [21], и они обнаружили, что у него есть серьезные ограничения. Было отмечено несколько неотъемлемых проблем, таких как синтезирование сети и данных атаки (после выборки фактического трафика) из-за проблем с конфиденциальностью, неизвестного числа отброшенных пакетов, вызванных переполнением трафика, и расплывчатыхопределений атаки. Tavallaee и другие также проводили статистические оценки и свои собственные эксперименты классификации. Они сообщили о большом количестве избыточных записей (78% в обучающих данных и 75% в тестовых данных), вызывающих смещение. Кроме того, в экспериментах классификации, проведенных группой, они отметили, **что случайным выбором подмножеств обучающих и тестовых данных часто может быть достигнута очень высокая нереалистичная точность**. Они предложили новый набор данных NSL-KDD, который состоит из выбранных записей полного набора данных KDD и не испытывает вышеупомянутых недостатков.

Набор DARPA 1998 определяет четыре типа атак: отказ в обслуживании (Denial of Service, DoS), получение прав root (User to Root, U2R), удаленный доступ к локальному компьютеру (Remote to Local, R2L) и проба или сканирование. DoS-атака - попытка отказать целевым пользователям или сетевым ресурсам. Атака U2R предоставляет root-доступ злоумышленнику. Атака R2L предоставляет доступ к локальной сети злоумышленнику. Атаки с помощью зондирования или сканирования собирают информацию о сетевых ресурсах. DARPA 1999 добавила новый тип атаки, в котором злоумышленник пытается **эксфильтровать** специальные файлы, которые должны остаться на компьютере-жертве.

**Методы машинного обучения и анализа данных для киберугроз  
(**IV. ML AND DM METHODS FOR CYBER **)**

В этом разделе описываются различные методы машинного обучения и анализа данных для кибербезопасности. Каждый метод **описывается некоторыми деталями**, и приводятся ссылки на цитируемые работы. Кроме того, для каждого метода представлены две, три статьи с их **применениями к киберсфере.**

**А. Искусственная нейронная сеть**

Искусственные нейронные сети (ИНС) вдохновлены человеческим мозгом и состоят из взаимосвязанных искусственных нейронов, способных к определенным вычислениям на их входах [22]. Входные данные активируют нейроны в первом слое сети, выход которых является входом во второй слой нейронов в сети. Аналогично, каждый уровень передает свой вывод на следующий уровень, а последний уровень выводит результат. Слои между входными и выходными слоями называются скрытыми слоями. Когда ИНС используется как классификатор, выходной уровень генерирует окончательную категорию классификации.

Классификаторы ИНС основаны на перцептроне [23] и были очень популярны до 1990-х годов, пока не были изобретены SVM. По сравнению с выпуклой квадратичной оптимизацией, применяемой в SVM, ИНС часто страдают от локальных минимумов и, таким образом, длительной работы во время обучения. В отличие от SVM, поскольку количество признаков в ИНС увеличивается, ее время обучения увеличивается. С одним или несколькими скрытыми слоями ИНС может генерировать нелинейные модели. Функция обратного распространения ошибки ИНС позволяет моделировать логику исключающего ИЛИ.

С разработками в этой области, такими как рекуррентные, прямого распространения и сверточные нейронные сети, ИНС снова становятся популярными, и в то же время выигрывая множество призов в последних конкурсах распознавания образов (эти соревнования еще не связаны с обнаружением кибервторжений). Поскольку расширенные версии ИНС требуют еще большей вычислительной мощности, они обычно реализуются на графических процессорах.

1) Обнаружение злоупотреблений: Cannady [24] использовал ИНС в качестве классификатора с несколькими категориями для обнаружения злоупотреблений. Он использовал данные, созданные сетевым монитором RealSecureTM, который имеет сигнатуры атак, встроенные в систему. Монитором было собрано десять тысяч событий, из которых 3000 - из смоделированных атак. Атаки были сымитированы программами Internet Scanner [25] и Satan [26].

На этапе предварительной обработки данных были отобраны девять признаков: идентификатор протокола (Protocol Identifier, ID), порт источника, порт назначения, адрес источника, адрес назначения, тип ICMP, ICMP-код, необработанная длина данных и необработанные данные. Десять процентов данных были выбраны случайным образом для тестирования. Затем исследование использовало оставшиеся нормальные данные и данные о нападениях для обучения ИНС, которая изучила комбинированные сигнатуры. В работе было выражено, что выводы были предварительными и сообщали о частоте ошибок при обучении и тестировании, которые были соответственно 0,058 и 0,070 среднеквадратичных (root-mean-square, RMS) ошибок. Хотя детали не были раскрыты, вывод ИНС был числом между 0 и 1, представляющим каждую из двух категорий (атака и нормальное состояние). Следовательно, среднеквадратичное значение 0,070 может быть примерно оценено как 93% точности для этапа тестирования. Каждый экземпляр пакета или данных был классифицирован как нормальная или атакующая группа.

2) Обнаружение аномалий и гибридное обнаружение: Lippmann и Cunningham [27] предложили систему, которая использует выбор ключевых слов и искусственные нейронные сети. Выбор ключевого слова выполнялся на расшифровках сеансов telnet, и статистика **была рассчитана на количество раз, когда каждое ключевое слово** (из предопределенного списка) **встречалось**. Статистика ключевых слов представляет собой вход в нейронную сеть, которая обеспечивает оценку апостериорной вероятности атаки. Вторая нейронная сеть работает с экземплярами, помеченными как атака, и пытается их классифицировать (т.е. предоставить имя атаки). Обе нейронные сети состояли из многослойных перцептронов без скрытых блоков. Система достигает 80% обнаружения с примерно 1 ложным сигналом тревоги в день. **Этот уровень ложной тревоги представляет собой улучшение на два порядка от базовой системы с той же точностью обнаружения.**

Bivens и другие [28] описывают полную IDS, которая использует этап предварительной обработки, кластеризацию нормального трафика, нормализацию, этап подготовки ИНС и этап принятия решения ИНС. На первом этапе использовалась самоорганизующаяся карта (Self-Organizing Map, SOM), которая является моделью неконтролируемой ИНС, для изучения шаблонов нормального трафика с течением времени, таких как обычно используемые номера портов TCP/IP. **Таким образом, первая стадия квантовала входные признаки в бункеры**, которые затем подавались на второй этап, многослойный перцептрон (Multilayer Perceptron, MLP) ИНС. Параметры сети MLP, такие как количество узлов и слоев, были определены первой ступени SOM. Как только обучение MLP было завершено, он начал предсказывать вторжения. Система может быть перезапущена для нового SOM, чтобы изучить новый шаблон трафика и для обучения нового классификатора MLP. В исследовании использовались данные TCP/IP из задачи DARPA 1999 [18], [19], где набор данных состоял из данных пакета сетевого уровня. В отличие от предыдущего исследования Cannady [24], которое классифицировало каждые данные пакетного уровня отдельно, эта система использовала временные окна для обнаружения и классификации группы пакетов. Таким образом, система смогла обнаружить типы атак **более длительной продолжительности**. Поскольку входные данные были низкоуровневыми сетевыми пакетными данными (в отличие от данных NetFlow), то степень детализации высокая, а полученные прогнозы все еще соответствуют короткой продолжительности. Bivens и другие [28] сообщили, **что успешно предсказывают 100% нормального поведения**. Их общий подход является многообещающим, хотя некоторые атаки не были полностью предсказаны, а FAR для некоторых атак достиг 76%.

**В. Ассоциативные правила и нечеткие ассоциативные правила**

Целью интеллектуального анализа ассоциативных правил является обнаружение ранее неизвестных ассоциативных правил из данных. Ассоциативное правило описывает взаимосвязь между различными атрибутами: ЕСЛИ (A И B) ТОГДА C. Это правило описывает отношение, что, когда присутствуют A и B, также присутствует C. В ассоциативных правилах есть метрики, которые сообщают, как часто данная связь возникает в данных. Поддержка - это предварительная вероятность (из A, B и C), а доверие - условная вероятность C, заданная A и B. Анализ ассоциативных правил был представлен Agrawal и другими [29] как способ обнаружить интересные совпадения в данных супермаркетов. Он находит часто встречающиеся наборы элементов (т.е. комбинации предметов, которые были приобретены вместе по крайней мере в N транзакциях в базе данных), а также из наборов часто встречающихся элементов, таких как {X, Y}, генерирует ассоциативные правила формы: X → Y И/ИЛИ Y → X.

**Рисунок 2**

Простой пример ассоциативного правила, относящегося к предметам, которые люди покупают вместе:

ЕСЛИ **(Хлеб** И **Масло) → Молоко (1)**

Это правило гласит, что, если человек покупает хлеб и масло, он также покупает молоко.

Ограничение традиционного анализа ассоциативных правил заключается в том, что он работает только с двоичными данными (т.е. элемент был приобретен либо в транзакции (1), либо нет (0)). Во многих реальных приложениях данные являются либо категориальными (например, имя IP, тип мероприятий в области здравоохранения), либо количественные (например, продолжительность, количество неудачных логинов, температура). Для числовых и категориальных атрибутов логические правила неудовлетворительны. Расширение, которое может обрабатывать численные и категориальные переменные, называется анализ нечетких ассоциативных правил [30].

Нечеткие ассоциативные правила имеют вид:

ЕСЛИ (X есть **A**) → (Y есть **B**) (2)

где X и Y - переменные, а A и B - нечеткие множества, характеризующие X и Y соответственно. Простым примером нечеткого ассоциативного правила для медицинского применения может быть следующее:

ЕСЛИ (температура - **Сильная лихорадка**) И (кожа - **Желтоватая**) И (потеря аппетита - **Глубокая**) → (**Острый** гепатит)

Правило гласит, что, если человек имеет **Сильную лихорадку**, Ж**елтоватую** кожу и **Глубокую** потерю аппетита, тогда у человека **Острый** гепатит. **Сильная лихорадка**, **Желтоватая**, **Глубокая** и **Острый** - это функции принадлежности переменных Температура, Кожа, Потеря аппетита и Гепатит, соответственно. В качестве примера нечетких функций принадлежности, функции принадлежности для переменной Температура показаны на **рисунке 2.**

Согласно определению на диаграмме, человек с температурой 100 °F имеет **Нормальную** температуру со значением принадлежности 0.2 и имеет **Лихорадку** со значением принадлежности 0.78. Использование нечетких функций принадлежности позволяет рассуждать с использованием лингвистических терминов. Эти лингвистические термины для температуры тела человека являются **Низкая**, **Нормальная**, **Лихорадка**, **Сильная** **лихорадка** и **Гипотермия**. Больше информации о нечеткой логике и нечетких функциях принадлежности можно найти в [31].

1) Обнаружение злоупотреблений: исследование Brahmi [32] является хорошим примером ассоциативных правил, применяемых к набору данных DARPA 1998, чтобы фиксировать отношения между параметрами TCP/IP и типами атак. В правилах, причины извлекаются из набора данных DARPA 1998, а следствия являются типами атак. Эта работа объясняет многомерный анализ ассоциативных правил, в котором в правилах существует несколько причин, таких как (ЕСЛИ (service И src\_port И dst\_port И num\_conn) ТОГДА attack\_type), что является примером четырехмерного правила. Работа включает в себя поиск правил с высокой поддержкой и высокой **уверенностью**. Наилучшая производительность достигается с использованием шестимерных правил с коэффициентами обнаружения 99%, 95%, 75%, 87% для типов атаки DoS, Probe или Scan, U2R и R2L, соответственно. Эксперименты не выполнялись с более высокими размерностями из-за вычислительных затрат. Одним из основных преимуществ анализа ассоциативных правил является то, что обнаруженные правила ясно объясняют отношения. Этот подход является перспективным для создания сигнатур **атаки**.

Zhengbing и другие [33] предложили новый алгоритм, основанный на **априорном алгоритме Подписи** [34] **для поиска новых сигнатур атак из существующих. Они сравнили время обработки своего алгоритма с временем подписи и обнаружили, что их алгоритм имеет более короткое время обработки, а разница в времени обработки увеличивается с увеличением размера базы данных.** Вклад этой статьи - описание нового метода поиска новых сигнатур атаки из существующих. Такой алгоритм может быть использован для получения новых сигнатур для включения в системы обнаружения злоупотреблений, такие как Snort [16].

2) Обнаружение аномалий и гибридное обнаружение: фреймворк NETMINE [35] выполняет обработку потока данных, **анализ уточнения** (путем сбора правил ассоциации из данных трафика) и классификации правил. Сбор данных выполняется одновременно с анализом потока онлайн. Пакеты трафика захватываются сетевыми инструментами захвата, разработанными в Туринском политехническом университете [36], работающими на основной линии сети кампуса. Захваченные данные являются данными NetFlow с такими атрибутами, как адрес источника, адрес назначения, порт назначения, порт источника и размер потока (в байтах).

NETMINE выполняет обобщенное извлечение ассоциативных правил для обнаружения аномалий и идентификации повторяющихся шаблонов. Индивидуальные ассоциативные правила (например, для одного конкретного IP-адреса) могут быть слишком подробными и иметь очень низкую поддержку. Обобщенные ассоциативные правила (например, для трафика подсети) позволяют повысить уровень абстракции, на котором представлены корреляции. Извлечение обобщенных правил выполняется новым алгоритмом Genio [37], который более эффективен, чем предыдущие подходы для добычи обобщенных правил. Процесс извлечения обобщенного ассоциативного правила не предназначен для выполнения в режиме реального времени во время сбора данных. **Однако некоторые эксперименты показывают возможность подхода к соответствующим частотам обновления скользящего окна.**

Классификация правил организует правила в соответствии с их семантической интерпретацией. Определены три основных класса правил:

* Правила потока трафика связаны с адресами источника и получателя.
* Предоставляемые правила служб состоят из порта назначения (т.е. сервиса) и адреса получателя (т.е. поставщика сервиса).
* Правила использования службы имеют порт назначения и адрес источника (т.е. пользователь сервиса).

Извлеченные правила призваны помочь сетевому аналитику в быстром определении шаблонов, которые заслуживают дальнейшего изучения. Автоматическая классификация отсутствует в нормальных и аномальных категориях.

**Работа Tajbakhsh и др. [38] использовала набор данных KDD 1999 для выполнения интеллектуального анализа нечетких ассоциативных правил для обнаружения общих шаблонов отношений. В исследовании использовалась исправленная версия набора KDD (см. Раздел III) с примерно 300000 экземпляров. Они использовали кластерный подход для определения нечетких функций принадлежности атрибутов, утверждая, что он работает лучше, чем подходы, основанные на гистограмме. Для сокращения элементов (согласно статье, имеется 189 различных элементов: 31 числовой атрибут с 3 значениями составляют 93 элемента и 10 номинальных атрибутов с общим количеством 96), в исследовании использовался метод гиперграничных ассоциаций. Например, набор элементов {a, b} считается гиперреберным, если средняя уверенность правил (a → b и b → a) больше порога. В работе используются пороговые значения в 98% для сокращения гиперграницы ассоциации и 50% для уверенности. Сообщается эффективность обнаружения аномалий со 100% точностью и коэффициентом FP 13%. Эффективность быстро падает, когда снижается коэффициент FP. В документе также говорится о преимуществах подхода интеллектуального анализа ассоциативных правил, например, о человеческих правилах, упрощении обработки символических (номинальных) атрибутов и эффективной классификации на больших наборах данных.**

Интеграция нечеткой логики с частотными эпизодами была предпринята Luo и Bridges [39], чтобы найти нечеткие частотные эпизоды, которые представляют частотные нечеткие последовательности в данных. **Фазификация** (т.е. квантование данных с перекрывающимися **ячейками**) помогает управлять числовыми переменными и объяснять переменные аналитику кибербезопасности. Частотные эпизоды определяются порогом, заданным пользователем. Эффективно, общий подход подобен **интеллектуальному анализу** последовательности. В эксперименте использовались данные, собранные tcpdump с сервера в университетском кампусе. Основные признаки из данных содержали флаги TCP и номера портов, а квантуются они нечеткой логикой. Моделирование вторжений проводилось специальными программами. В исследовании сообщается о измерениях сходства между наборами обучающих и тестовых данных. Наибольшее зарегистрированное сходство составляет 0.82. Хотя в работе не сообщалось о показателях эффективности, общий подход является многообещающим, тем самым расширяя и улучшая предыдущие подходы в литературе.

**С. Байесовская сеть (Bayesian Network)**

Байесовская сеть является вероятностной графической моделью, которая представляет переменные и отношения между ними [40], [41]. **Сеть построена с узлами, в качестве дискретных или непрерывных случайных переменных и направленными ребрами как отношениями между ними, создавая направленный ациклический граф.** Дочерние узлы зависят от их родителей. Каждый узел поддерживает состояния случайной величины и форму условной вероятности. Байесовские сети строятся с использованием экспертных знаний или с использованием эффективных алгоритмов, которые выполняют логический вывод.

На рисунке 3 приведен пример байесовской сети для обнаружения сигнатур атаки.

**Рисунок 3. Пример Байесовской сети для обнаружения сигнатур.**

Каждое состояние (или сетевая переменная) может быть входом в другие состояния с определенным набором значений состояния. Например, состояние протокола может выбирать значения из доступных номеров протоколов. Каждое из значений состояния, которое может перейти из одного состояния в другое, имеет **связанную вероятность**, **и сумма этих вероятностей будет содержать до 1**, представляющую полный набор значений состояния. В зависимости от применения, сеть может использоваться для объяснения взаимодействия между переменными или для вычисления вероятного результата для целевого состояния (например, предупреждения или доступа к файлам) с использованием входных состояний.

Таблицы вероятности могут быть рассчитаны из имеющихся обучающих данных. Вывод ненаблюдаемых переменных, изучение параметров и изучение структуры являются одними из основных задач обучения байесовских сетей.

1) Обнаружение злоупотреблений: В общем случае обнаружение аномалий может считаться реагирующим, потому что система откликается на вход, когда ввод является неожиданным. И наоборот, в проблеме обнаружения злоупотреблений система является упреждающей, потому что сигнатуры, извлеченные из ввода, постоянно проверяются на список шаблонов атаки. Взятие упреждающего подхода, как и при обнаружении злоупотреблений, требует классификации сетевых потоков. В своей работе, Livadas и другие [42] сравнили несколько методов, применяемых к проблеме DoS. Их работа пыталась разрешить трафик ботнета в отфильтрованном трафике интернет-ретрансляции (Internet Relay Chat, IRC), следовательно, определяя существование ботнета и его происхождение. В исследовании использовались данные уровня TCP, собранные из 18 мест в беспроводной кампусной сети Дартмутского университета в течение 4 месяцев. Для создания сетевых потоков или данных NetFlow используются данные TCP (основанные на IRC бот-сети не используют UDP или ICMP). Фильтрующий слой используется для извлечения данных IRC из всех сетевых данных.

Поскольку **основную истину** трудно получить, в исследовании использовались определенные аспекты коммуникации IRC для маркировки данных с помощью ярлыков класса IRC. Помечать генерируемые данные IRC-ботнета еще сложнее, поэтому в исследовании используются смоделированные данные для экспериментов. Производительность байесовской сети оценивается с точностью 93% и коэффициентом FP ​​1.39%. Два других классификатора, Наивный Байесовский и дерево решений C4.5, достигают точности 97%. Тем не менее, их коэффициент FP выше - 1,47% и 8,05% соответственно.

В другом исследовании, Jemili и другие [43] предложили структуру IDS, использующую байесовские сетевые классификаторы. В работе использовались девять признаков данных KDD 1999 в сети вывода.На этапе обнаружения аномалий решение о нормальном или атаке принимается модулем вывода дерева соединений с эффективностью 88% и 89% по нормальному и атакующему трафику соответственно. На следующем этапе типы атак были распознаны из данных, помеченных как данные атак, модулем обнаружения аномалий. Эффективность в 89%, 99%, 21%, 7% и 66% сообщаются для DoS, Probe или Scan, R2L, U2R и других классов соответственно. Исследование предполагает низкую эффективность категорий R2L и U2R из-за того, что количество учебных случаев намного ниже, чем для других категорий.

2) Обнаружение аномалий и гибридное обнаружение: Когда вычислительная платформа получает пакеты TCP/IP, сетевой стек базовой ОС обрабатывает эти пакеты. Сетевой стек генерирует различные журналы и вызовы ядра системы, и, в конечном итоге, пакетные данные обрабатываются на уровне приложения, вызываемого ядром. Kruegel и другие [44] использовали байесовскую сеть для классификации событий во время открытых и исполнительных вызовов ОС. Набор данных DARPA 1999 используется для возбуждения ядра ОС с помощью пакетов TCP/IP. Затем сравнивается набор атрибутов, основанных на этих системных вызовах, таких как длина строки аргумента системного вызова, распределение и распределение символов, с использованием теста Пирсона. Кроме того, в строке системных вызовов выполняется поиск структуры строки по отношению к грамматике (синтаксису команды) и токенам. Эти функции используются в байесовской сети для расчета вероятности нормального или аномального состояний. Поскольку пороговое значение обнаружения используется для управления FAR, система является гибкой и может выполнять самостоятельную корректировку против слишком большого количества ложных тревог; 75% точность и 0,2% FAR, 100% точность и 0,1% FAR достигаются с использованием разных пороговых значений.

Определение сложных атак часто требует наблюдения аномалий и корреляций между ними, чтобы выявить сценарии или планы нападений. Для достижения этой цели была исследована корреляция тревог, чтобы уменьшить объем предупреждений, генерируемых системой. Кластеризация, меры сходства или экспертные знания используются для корреляции аномалий, а затем выявляют сложные шаблоны сценариев атаки.

Детектор вторжений DoS, который использует байесовскую сеть, описывается Benferhat и другими [45]. Существует только один родительский узел, представляющий скрытую переменную (т.е. класс - {нормальный, аномальный}), а наблюдаемые переменные являются дочерними узлами. Предполагается, что дочерние узлы статистически независимы. Основная цель этого подхода - выполнять корреляцию сигналов, используя исторические данные с минимальным использованием экспертных знаний. В экспериментах используется набор данных DARPA 2000. Для каждой цели вторжения построена сеть. Установки исследования имеют два разных сценария, извлеченных из набора данных DARPA; Система успешно обнаружила один из сценариев, но не обнаружила другого. К сожалению, в исследовании не сообщается никаких количественных результатов.

**D. Кластеризация (Clustering)**

Кластеризация [46] - это набор методов для поиска шаблонов в высокоразмерных немаркированных данных. Это неконтролируемый подход обнаружения шаблонов, в котором данные группируются вместе на основе меры сходства. Основным преимуществом кластеризации для обнаружения вторжений является то, что она может учиться на основе данных аудита, не требуя от системного администратора предоставлять явные описания различных классов атак.

Существует несколько подходов для кластеризации входных данных. В моделях подключения (например, иерархическая кластеризация) точки данных группируются по расстояниям между ними. В центроидных моделях (например, k-средних) каждый кластер представлен его средним вектором. В моделях распределения (например, алгоритм максимизации ожиданий) предполагается, что группы согласуются со статистическим распределением. Модели плотности группируют точки данных как плотные и связанные области (например, пространственная кластеризация приложений с шумом на основе плотности (Density-Based Spatial Clustering of Applications with Noise, DBSCAN)). Наконец, графические модели (например, CLIQUE) определяют каждый кластер как набор связанных узлов (точек данных), где каждый узел имеет ребро по меньшей мере для одного другого узла в наборе.

Алгоритм обучения на основе экземпляров (также называемый ленивым обучением), k-NN, является другим популярным методом ML, где классификация точки определяется k ближайшими соседями этой точки данных. Пример такого подхода для кибербезопаности описан в [47]. Определение категории точки осуществляет **большинство голосов**, что может быть недостатком, если распределение классов искажено. Поскольку высокоразмерные данные отрицательно влияют на k-NN методы (т.е. проклятие размерности), почти всегда требуется **сокращение признаков**. Также считается, что выбор порядка соседства или порядка в наборе данных имеет большое влияние. К преимуществам k-NN относится его простота и отсутствие параметрических допущений (кроме числа k).

В теории графов, коэффициент кластеризации представляет собой родственность узлов, которые близки друг к другу [48]. Некоторые типы данных, такие как социальные сети, имеют высокие коэффициенты кластеризации. Глобальный коэффициент кластеризации определяется как отношение числа замкнутых триплетов к числу связанных триплетов вершин. Локальный коэффициент кластеризации узла определяется как отношение числа подграфов с тремя ребрами и тремя вершинами, которые являются частью числа троек, частью которых является узел [49]. Коэффициент находится между 0 и 1, 0 - один узел, а 1 - каждый сосед, соединенный с узлом.

1) Обнаружение злоупотребления: в литературе существует меньше применений методов кластеризации для обнаружения злоупотребления, чем для обнаружения аномалий. Однако, как показано в одном из исследований [50], генерация сигнатур в реальном времени детектором аномалий может быть важным активом. Для создания кластеров нормального и вредоносного сетевого трафика используется схема кластеризации на основе плотности, называемая Simple Logfile Clustering Tool (SLCT). Чтобы различать нормальный и аномальный трафик, в исследовании использовался параметр M, чтобы установить процент фиксированных признаков, которые содержит кластер. **Фиксированный признак для кластера соответствует постоянному значению для этого признака. Исследование определяет, как плотную одну область значения признаков, которые по крайней мере в N процентах экземпляров.** Это, по существу, значение поддержки одного антецедента, одна из метрик, которые использует интеллектуальный анализ ассоциативных правил. Таким образом, когда M равно нулю, все данные кластеризованы, а когда M имеет высокое значение, в идеале остаются только вредоносные кластеры. Например, установив значение M на 97%, исследование обнаруживает 98% данных атаки с 15% FAR. Метод может обнаруживать ранее невидимые (новые или нулевого дня) атаки. После этапа кластеризации с указанными параметрами все кластеры рассматриваются как атаки, которые обрабатывают кластерные центроиды в качестве сигнатур.

Система состоит из двух схем кластеризации: первая используется для обнаружения нормального или атакующего трафика (как описано прежде), а вторая используется контролируемым образом для определения обычного трафика. Разница между ними заключается в изменении настройки параметров, практически имея две схемы кластеризации для обнаружения нормального и аномального трафика параллельно. Выход этого этапа переходит к модулю на основе правил для извлечения сигнатур, которые будут использоваться либо системой, либо специалистами в области кибербезопасности. Одна из новинок в этом исследовании заключается в том, что каждый аномальный кластерный центроид рассматривается как сигнатура, подлежащая отфильтровыванию системой (например, после ежечасного или ежедневного обновления сигнатур системой).

В исследовании использовался набор данных KDD в нескольких экспериментах. Наборы данных были подготовлены с процентами атак 0%, 1%, 5%, 10%, 25%, 50% и 80%. Метрики производительности, такие как целостность кластера, использовались в дополнение к точности. Исследование показало, что их эффективность составляет 70% -80% -ный показатель обнаружения для ранее неизвестных атак. Результаты впечатляют, особенно учитывая тот факт, что система не имела никаких предварительных знаний об атаках или шаблонах атаках в наборе данных KDD.

2) Обнаружение аномалий и гибридное обнаружение: В своем исследовании Blowers и Williams [51] использовали метод кластеризации DBSCAN для группировки нормальных и аномальных сетевых пакетов. Набор данных KDD предварительно обрабатывается для выбора признаков с использованием корреляционного анализа. Хотя фактический FAR не сообщается, в исследовании используется порог метода кластеризации для управления FAR системы. При предварительной обработке данных устанавливается отношение 10% атак к отсутствию атак. Сообщаемая производительность составляет 98% для обнаружения атаки или отсутствия атаки. Это очень высокое значение для аномального детектора, основанного на кластеризации, выше, чем в большинстве исследований в литературе. В целом, исследование представляет собой хороший пример **подведения итогов применения** методов ML для киберопераций.

Sequeira и Zaki [52] взяли 500 сеансов с длинным потоком команд от девяти пользователей в Университете Пердью. Данные пользовательского командного уровня (команды оболочки) использовались для определения того, является ли пользователь обычным пользователем или злоумышленником. Командный поток каждого пользователя в сеансе анализировался в токенах и впоследствии представлялся в виде последовательности токенов. Sequeira и Zaki пытались использовать несколько подходов, включающих последовательности, такие как последовательности мер сходства и алгоритмы сопоставления последовательностей. Один из перспективных подходов был основан на самой длинной общей метрике подпоследовательности.

Команда, взятая у девяти пользователей, предполагает, что данные малы, возможно, возможно из-за требований к обработке. Неясно, сколько команд содержит типичный сеанс, но есть хорошая вероятность, что он намного длиннее максимальной длины последовательности, которую исследование выбрало как 20. Исследование показало, что его производительность составляет 80% точности с 15% FAR **в качестве успешного результата**. Sequeira и Zaki [52] ссылаются на предыдущие исследования, которые использовали один и тот же набор данных и достигли 74% с 28% FAR, и указывают на улучшение. Также может быть хорошей идеей использовать **журналы сервера DARPA или KDD для сравнения.**

**E. Деревья решений (Decision Trees)**

Дерево решений - это древовидная структура, у которой есть листья, которые представляют собой классификации, и ветви, которые, в свою очередь, представляют собой сочетания признаков, которые приводят к этим классификациям. Пример обозначается (классифицируется) путем тестирования его признаков (атрибутов) на узлах дерева решений. Наиболее известные методы автоматического построения деревьев решений - это алгоритмы ID3 [53] и C4.5 [54]. Оба алгоритма строят деревья решений из обучающего набора данных, используя концепцию информационной энтропии. При построении дерева решений на каждом узле дерева C4.5 выбирает атрибут данных, который наиболее эффективно разбивает его набор примеров на подмножества. Критерием разбиения является нормализованный прирост информации (разница в энтропии). Для принятия решения выбирается атрибут с самым высоким нормализованным приростом информации. Затем алгоритм C4.5 выполняет рекурсию на меньших подмножествах, пока все учебные примеры не будут классифицированы.

Преимущества деревьев решений - это интуитивное выражение знания, высокая точность классификации и простая реализация. Основным недостатком является то, что для данных, включающих категориальные переменные с различным числом уровней, значения прироста информации смещены в пользу признаков с большим числом уровней. Дерево решений построено путем максимизации прироста информации при каждом разделении переменных, что приводит к естественному ранжированию переменных или выбору признаков. Маленькие деревья (например, изображенные на Рис. 4) имеют интуитивное выражение знания для экспертов в данной области, потому что легко извлекать правила из этих деревьев, просто изучая их. Для более глубоких и более широких деревьев гораздо труднее извлечь правила и, следовательно, чем больше дерево, тем **менее интуитивно выражение его знания**. Меньшие деревья получают из более крупных путем обрезания. Большие деревья часто имеют высокую точность классификации, но не очень хорошие возможности обобщения. Обрезая большие деревья, получают более мелкие деревья, которые часто имеют лучшие возможности обобщения (они избегают переобучения). Алгоритмы построения дерева решений (например, C4.5) относительно просты, чем более сложные алгоритмы, такие как SVM. Таким образом, они также имеют более простую реализацию.

**Рисунок 4. Пример дерева решений**

1) Обнаружение злоупотребления: большинство систем обнаружения злоупотреблений выполняют обнаружение путем сравнения каждого входа со всеми правилами (сигнатурами). Snort [16], известный инструмент с открытым исходным кодом, следует подходу, основанному на сигнатурах. В Snort каждая сигнатура имеет однострочное описание. Процесс сопоставления между входными данными и сигнатурами обычно медленный (особенно когда число велико) и поэтому неэффективно для использования в пользовательском интерфейсе.

Kruegel и Toth [55] заменили механизм обнаружения злоупотреблений Snort деревьями решений. Сначала они выполнили кластеризацию правил, используемых Snort 2.0, а затем вывели дерево решений, используя вариант алгоритма ID3. Кластеризация правил минимизирует число сравнений, необходимых для определения того, какие правила запускаются с помощью заданных входных данных. Дерево решений выбирает наиболее отличительные признаки набора правил, что позволяет проводить параллельную оценку каждого признака. Это дает производительность, намного превосходящую производительность Snort.

Предложенный метод был применен к файлам tcpdump из тестовых данных за 10 дней, полученных MIT/LL для оценки обнаружения вторжений DARPA 1999. Для этого набора данных сравнивалась скорость работы Snort и метод дерева решений. Фактическое увеличение производительности значительно варьируется в зависимости от типа трафика; максимальное ускорение составило 105%, в среднем 40,3% и минимум 5%. Эксперименты проводились также путем увеличения количества правил от 150 до 1581 (полный набор, используемый Snort 2.0). С увеличением количества правил ускорение метода дерева решений над Snort 2.0 еще более выражено.

Это исследование показало, что методы кластеризации в сочетании с деревьями решений могут существенно сократить время обработки системы обнаружения злоупотреблений, возможно, позволяя им эффективно использоваться в пользовательском интерфейсе.

2) Обнаружение аномалий и гибридное обнаружение: EXPOSURE [56], [57] - это система, которая использует широкомасштабные пассивные методы анализа доменных имен (Domain Name Service, DNS) для обнаружения доменов, которые вовлечены в злонамеренную деятельность. Система состоит из пяти основных компонентов: сборщика данных, компонента атрибуции признаков, сборщика вредоносных и доброкачественных доменов, модуля обучения и классификатора. **Классификатор построен по дереву решений дерева Weka J48**, который является реализацией алгоритма C4.5, способного генерировать обрезанные или необрезанные деревья решений. Эксперимент показал, что минимальная ошибка была получена, когда все признаки были объединены, и поэтому все 15 признаков использовались деревом решений.

Используемые данные состоят из данных DNS, собранных за 2,5 месяца (100 миллиардов DNS-запросов, что привело к 4,8 миллионам отдельных доменных имен). **Исследование изучило** несколько тысяч вредоносных и доброкачественных доменов и использовало их для построения обучающего набора. Вредоносные домены были получены из www.malwaredomains.com, списка блокировок Zeus, Anubis и т. Д. Первоначальный список вредоносных доменов состоит из 3500 доменов. Доброкачественные домены были из списка Alexa топ 1000 лучших доменов.

Экспериментируя с разными значениями длин периодов, исследование определило, что оптимальный период начального обучения для системы составил 7 дней. После этого начального обучения классификатор переучивался каждый день. Результаты значительно варьируются в зависимости от набора данных. В целом, используя десятикратную перекрестную проверку, точность обнаружения вредоносных доменов составила 98,5%, а FAR – 0,9%.

Были проведены дополнительные эксперименты, чтобы определить, может ли метод обнаруживать вредоносные домены, отсутствовавшие в обучающем наборе. В первом эксперименте, среди 50 случайно выбранных доменов из списка 17686 доменов, классификатор обнаружил, что три доброкачественных домена являются злонамеренными (6% FP rate). Во втором эксперименте, исследование автоматически перекрестно проверяет вредоносные и подозрительные домены, идентифицированные классификатором, с использованием инструментов оценки веб-сайтов, таких как McAfee Site Advisor, Google Safe Browsing и Norton Safe Web. Для выявленных вредоносных доменов **коэффициент** FP составил 7,9%. В третьем эксперименте EXPOSURE классифицировал 100 миллионов DNS-запросов в течение 2-недельного периода, обнаружил 3117 новых вредоносных доменов (ранее не известных системе и не используемых в обучении), и за это время не генерировал никаких FP.

Экспериментальные результаты показывают, что точность и FAR системы EXPOSURE являются удовлетворительными и что EXPOSURE полезна при автоматической идентификации разнообразной категории вредоносных доменов (например, серверы управления и управления ботнетом, фишинговые сайты, хосты для скама). Важным достижением является способность EXPOSURE обнаруживать большое количество ранее неизвестных вредоносных доменов из DNS-трафика.

**F. Обучение ансамбля (Ensemble Learning)**

В общем, алгоритмы обучения с учителем ищут пространство гипотез для определения правильной гипотезы, которая будет давать хорошие прогнозы для данной проблемы. Хотя хорошие гипотезы могут существовать, может быть трудно найти **их**. **Методы ансамбля объединяют множество гипотез, надеясь сформировать лучшую, чем лучшая гипотеза**. Часто методы ансамбля используют несколько слабых учеников для создания сильного ученика [58].

Слабый ученик - это тот, который последовательно генерирует лучшие прогнозы, чем случайные. Один из алгоритмов ансамбля использует усиление для тренировки нескольких слабых алгоритмов обучения и объединения (т.е. суммирования) их взвешенных результатов. Adaptive Boosting (AdaBoost) [59] - один из наиболее популярных алгоритмов, используемых для уменьшения проблемы переобучения, присущей ML. Ускорение можно рассматривать как линейную регрессию, в которой признаки данных являются вкладом слабого ученика h (например, прямая линия, делящая входные точки данных на две категории в пространстве), а выход усиления - это взвешенное суммирование этих h функции. **Long и Servedio [60] критиковали усиление, говоря о том, что ненулевая доля неправильно маркированных данных может стать причиной того привести к сбою в сбое полностью и привести к ROC 0.5 (что эквивалентно случайной догадке).**

Бэггинг (**bootstrap aggreagation**) - это метод улучшения обобщенности предсказательной модели для уменьшения переобучения. Он основан на методе усреднения модели и, как известно, улучшает эффективность кластеризации 1-ближайшего соседства.

**Классификатор Случайный лес** [61] - метод ML, который объединяет деревья принятия решений и обучение ансамбля. Лес состоит из многих деревьев, которые используют случайно выбранные признаки данных (атрибуты) в качестве их входа. Процесс генерации леса создает коллекцию деревьев с регулируемой дисперсией. Полученный прогноз может быть решен путем голосования по большинству голосов или взвешенного голосования.

Случайные леса имеют несколько преимуществ: низкое количество **параметров управления и модели**; устойчивость к переобучению; нет необходимости в выборе признаков, поскольку они могут использовать большое количество возможных атрибутов. Одним из важных преимуществ случайного леса является то, что дисперсия модели уменьшается по мере увеличения количества деревьев в лесу, тогда как смещение остается неизменным. Случайные леса также имеют некоторые недостатки, такие как низкая интерпретация модели, потеря производительности из-за коррелированных переменных и зависимость от случайного генератора реализации.

1) Обнаружение злоупотреблений: Zhang и др. [62] подходят к проблеме обнаружения злоупотреблений, используя модуль обнаружения выбросов **на внешнем интерфейсе** своей системы. Если ввод классифицирован как ненормальный сетевой трафик, данные далее классифицируются как принадлежащие к одной из категорий атак набора данных KDD 1999. В исследовании представлено полное системное решение, включающее детектор выбросов, предиктор **атак на основе сигнатур** и базу данных шаблонов. База данных аномалий также используется для хранения шаблонов, помеченных как аномалия либо пользователем (вручную), либо системой (автоматически) с использованием предварительно помеченных данных. Параметры случайного леса определяются и оптимизируются путем использования разных значений для сбалансированного обучающего набора. Исследование создало сбалансированный набор данных путем повторения наименее встречающихся экземпляров атаки, которые могут считаться неправильным подходом. Не существует стадии проверки, которая не рекомендуется, поскольку параметры системы определяются данными из обучающего набора, что уменьшает обобщение модели. Тем не менее, общий подход **является обоснованным**, поскольку он успешно использует полезные свойства случайных лесов.

Исследование сгруппировало набор данных с атаками (DoS, Probe или Scan) и **атаками меньшинства** (U2R и R2L). Сообщаемая эффективность при обнаружении злоупотреблений составила 1.92% ошибки в исходном наборе и 0,05% на сбалансированном наборе с 0,01% и 0% FAR соответственно. Исследование также сгруппировало атаки DoS и Probe или Scan на четыре уровня атаки (1%, 2%, 5% и 10%), которые представляют собой процент данных атаки в тестовом наборе. Точность, достигнутая для этих четырех уровней атаки, составляет 95%, 93%, 90% и 87%, соответственно, с постоянной 1% FAR. Обнаружение меньшинства (U2R и R2L) сообщается как 65% с 1% FAR. Реализация является достаточно быстрой, чтобы использоваться в качестве онлайн-решения.

В другом исследовании [63] использовался случайный лес для обнаружения злоупотреблений на том же наборе данных KDD 1999, в стиле, подобном Zhang и др. [62]. Основное различие заключается в том, что этап нормализации происходит в модуле предварительной обработки. Это исследование сравнивает эффективность случайных лесов с Гауссовским максимальным правдоподобием, **наивными байесами** и классификаторами дерева решений. Точность, о которой сообщалось для случайного леса, составляет 97%, 76%, 5% и 35% для DoS, Probe или Scan, R2L и U2R, соответственно.

Ансамбль из трех ИНС (см. Подраздел A, один SVM (см. Подраздел L) и один многовариантный адаптивный регрессионный сплайн (Multivariate Adaptive Regression Spline, MARS) [64] используется Mukkamala и др. [65] для обнаружения вторжений. Первая ИНС является эластичной нейронной сетью обратного распространения, вторая является ИНС с масштабируемым сопряженным градиентом, а третья – ИНС с одношаговым секущим алгоритмом. **Мажоритарное голосование** используют для объединения результатов пяти классификаторов.

Используемые данные являются подмножеством набора данных DARPA 1998. Это подмножество состоит из 80% атак и 20% от нормальных данных. Показатели 99,71%, 99,85%, 99,97%, 76% и 100% представлены для категорий Normal, Probe или Scan, DoS, U2R и R2L соответственно. Метод ансамбля превосходит точность отдельных классификаторов, входящих в ансамбль. Точность четырех классов находится в диапазоне 99%; Только точность в классе U2R намного ниже на 76%.

2) Обнаружение аномалий и гибридное обнаружение: система DISCLOSURE от Bilge и др. [66] выполняет обнаружение бот-сети Command and Control (C & C) с использованием случайных лесов. Она использует данные NetFlow, которые легко доступны, но включает только совокупные метаданные, связанные с продолжительностью потока и количеством переданных пакетов, а не полной полезной загрузкой пакетов. Признаки, основанные на потоке, извлекаются из данных (например, размер потока, шаблоны доступа клиентов, временные признаки). В их статье нет данных о том, сколько деревьев использовалось в классификаторе случайного леса или сколько атрибутов использовалось деревьями в среднем. DISCLOSURE была протестирована в двух реальных сетях и достигла коэффициента True Positive (обнаружение) 65%, коэффициента False Positive – 1%.

Применение случайных лесов для обнаружения аномалий описано Zhang и др. [62], где детектор аномалий (выброса) использовался для подачи второго классификатора угроз. В сущности, метод является гибридным, используя два классификатора случайного леса - один для обнаружения аномалий, а другой для обнаружения злоупотреблений. Эффективность обнаружения выброса составила 95% точности с 1% FAR. В исследовании показано, как детектор аномалий может быть реализован с использованием меры близости (т.е. расстояние между двумя экземплярами, вычисленными из леса) случайного леса. Он использовался как **мера суммы квадратов близости** между деревьями и категориями.

**G. Эволюционные вычисления (Evolutionary Computation)**

Термин эволюционные вычисления включает в себя генетические алгоритмы (Genetic Algorithms, GA) [67], генетическое программирование (Genetic Programming, GP) [68], стратегии эволюции [69], оптимизацию роевых частиц [70], оптимизацию колоний муравьев [71] и искусственные иммунные системы [72]. В этом подразделе основное внимание уделяется двум наиболее широко используемым эволюционным вычислительным методам - GA и GP. Они оба основаны на принципах выживания наиболее приспособленных. Они работают на популяции особей (хромосом), которые развиваются, используя определенные операторы. Основными операторами являются отбор, скрещивание и мутация. Обычно они начинаются со случайно сгенерированной популяции. Для каждой особи из популяции вычисляется значение пригодности, которое показывает, насколько хорошо данная особь решает проблему. Особи с более высокой пригодностью имеют более высокую вероятность быть выбранными в родительский пул и, таким образом, способны размножаться. Две особи из родительского пула могут выполнить скрещивание (т.е. обменяться генетическим материалом между ними), и каждая из них также может подвергнуться мутации, которая является случайным изменением генетического материала индивидуума. Особи с наивысшей пригодностью копируются в следующее поколение.

Основное различие между GA и GP заключается в том, как особи представлены. В GA они представлены как битовые строки, а операции скрещивания и мутации очень просты. В GP особи представляют программы и, следовательно, представляют деревья с операторами, такие как *плюс*, *минус*, *умножение*, *деление*, *ИЛИ*, *И*, *НЕ,* или даже программирующие блоки, такие как, *ЕСЛИ...ТОГДА*, *цикл* и т.д. В GP операторы скрещивания и мутации гораздо сложнее, тем те, которые используются в GA.

1) Обнаружение злоупотреблений: Li [73] разработал метод, который использует GA для разработки правил обнаружения злоупотребления. Использовался набор данных обнаружения вторжений DARPA. Хромосома GA предназначена для размещения IP-адреса источника, IP-адреса получателя, номера порта источника, номера порта получателя, продолжительности соединения, протокола, количества байтов, переданных отправителем и ответчиком, и состояния соединения. В функции пригодности соглашение по определенным частям хромосомы (например, IP-адрес получателя) взвешивается выше, чем у других (например, тип протокола). Традиционные операторы скрещивания и мутации применяются к особям из популяции. Методы «нишинга» используются для нахождения множественных локальных максимумов (потому что требуется множество правил, а не только одно). Лучшие разработанные правила становятся частью базы правил для обнаружения вторжений. Хотя статья Li описывает интересный метод и показывает некоторые правила, разработанные системой, что отсутствует, несмотря на 181 цитирование работы, так это точность набора правил на тестовых данных.

Abraham и другие [74] используют GP для разработки простых программ классификации атак. Три метода GP, используемые в экспериментах, являются линейным генетическим программированием (Linear Genetic Programming, LGP), программирование с несколькими выражениями (Multi-Expression Programming, MEP) и программирование экспрессии генов (Gene Expression Programming, GEP). **В рамках этой программы в качестве набора функций использовались +, -, \*, /, sin, cos, sqrt, ln, lg, log2, min, max и abs.** Различные подмножества признаков используются различными методами GP. Использовался набор данных обнаружения вторжений DARPA 1998. Набор данных содержит 24 типа атаки, которые можно разделить на четыре основные категории: DoS, несанкционированный доступ с удаленного компьютера (R2L), несанкционированный доступ к локальному суперпользователю (U2R), а также наблюдение и прочее зондирование (проба или сканирование). FAR варьируется от 0% до 5,7% в зависимости от используемого метода и типа атаки.

Hansen и другие [75] использовали GP с гомологичным скрещиванием для разработки программ, осуществляющих обнаружение вторжений. Гомологичное скрещивание - это специальный оператор скрещивания, призванный уменьшить тенденцию развития программ, которые все больше возрастают с увеличением числа поколений. Использовалось подмножество данных KDD 1999, в котором было задействовано 30 000 экземпляров для обучения и 10 000 для тестирования. Наборы данных для обучения и тестирования были выбраны таким образом, что они имеют одинаковые пропорции атак каждого типа, как полный набор данных. Их точность варьировалась от 66,7% до 100% в зависимости от типа атаки (самая низкая для U2R и самая высокая для DoS). Специфичность (таблица V) варьировалась от 99,56% до 100%, что означает, что FAR очень низок. GP с гомологичными результатами скрещивания также сравниваются с результатами победителя KDD 1999 Cup, и они лучше. Однако, это сравнение может быть недействительным, поскольку результаты в этой работе предназначены только для выбранного тестового набора в 10 000 сигнатур, а результаты для победителя KDD Cup могли быть для другого набора тестовых данных.

2) Обнаружение аномалий и гибридное обнаружение: Khan [76] использует GA, чтобы разработать правила для обнаружения вторжений. Использовались два подмножества, каждый из 10 000 экземпляров из набора данных KDD 1999: один для обучения и один для тестирования. **Для большого количества атрибутов в этих данных**, восемь были выбраны с использованием Метода главных компонент [77]. Была использована популяция из 10 особей, **и, по-видимому, для классификации данных было использовано только одно правило из конечной популяции для разделения данных на два класса: нормальный и атакующий.** Удивительно, что результаты, полученные в наборе тестовых данных, на самом деле лучше, чем результаты, полученные в наборе данных обучения. В точности и FARs являются 93,45% (10,8% ВСР) и 94,19% (2,75% ВСР) для нормального и атакующего классов, соответственно.

Lu и Traore [78] представили подход эволюции правил, основанный на GP для обнаружения известных и новых атак в сети. Исходная популяция правил была выбрана на основе базовых знаний известных атак, и каждое правило может быть представлено как дерево разбора. GP развивает эти первоначальные правила, чтобы генерировать новые правила, используя четыре генетических оператора: воспроизведение, скрещивание, мутацию и **условный оператор сброса**. **УсОператор условия сброса** случайным образом выбирает одно условие в правиле, и затем это условие больше не рассматривается в правиле, тем самым препятствуя тому, чтобы программы становились все более сложными с количеством поколений. Используемая функция пригодности основывалась на поддержке и уверенности в правиле (см. Подраздел 1). В работе используется подмножество данных обнаружения вторжений DARPA. Обучающий набор данных состоит из записей соединения за 1 день (т.е. 10 000 записей соединений) с восемью типами атак. Набор данных тестирования состоит из записей соединения за другой день с 10 типами атак (два типа атаки являются новыми). В практической оценке, база правил вместо одного правила используется для проверки производительности IDS. Выполняются десять тысяч прогонов GP, и сообщаются средние результаты. Среднее значение FAR составляет 0,41%, а среднее значение PD составляет 0,5714. ROC показывает PD около 100%, когда FAR находится в диапазоне от 1.4% до 1.8%. Однако, когда FAR близок к 0%, PD составляет лишь около 40%. PD находится в широком диапазоне от 40% до 100%, потому что количество правил в базе правил различно для каждого прогона. Результаты сообщаются по данным, которые содержат новые атаки, а также известные атаки. В отличие от многих других работ, которые приводят только лучшие результаты, Lu и Traore включают средние результаты. Их наилучшие результаты, с PD около 100% и FAR в диапазоне от 1,4% до 1,8% **являются хорошими**.

**H. Скрытые Марковские модели (Hidden Markov Models)**

Марковские цепи и скрытые Марковские модели (Hidden Markov Models, HMM) принадлежат к категории Марковских моделей. Цепь Маркова [79] является набором состояний, взаимосвязанных через вероятности переходов, которые определяют топологию модели. HMM [80] является статистической моделью, **где предполагается, что система, являющаяся моделируемой, считается Марковским процессом** с неизвестными параметрами. Основная задача состоит в том, чтобы определить скрытые параметры из наблюдаемых параметров. Состояния HMM представляют собой ненаблюдаемые условия, являющиеся моделируемыми. Имея разную выходную вероятность распределений в каждом состоянии и позволяя системе изменять состояния со временем, модель способна представлять нестационарные последовательности.

Пример HMM для обнаружения вторжений на хост показан на рисунке 5 [81]. В этом примере, каждый хост моделируется четырьмя состояниями: *Хороший*, *Зондированный*, *Атакованный*  и *Скомпрометированный*. Ребро от одного узла к другому представляет тот факт, что, когда хост находится в состоянии, указанном исходным узлом, он может перейти к состоянию, указанному **узлом-получателем**. **Матрица вероятностей переходов состояний** P описывает вероятности переходов между состояниями модели. Матрица вероятностей наблюдений Q описывает вероятности получения различных наблюдений, учитывая, что хост находится в определенном состоянии. π - начальное распределение состояний. HMM обозначается (P, Q, π).

1) Обнаружение злоупотреблений: в исследовании Ariu и др. [82] рассматриваются атаки на веб-приложения (такие как XSS и SQL-Injection), а HMM используются для извлечения сигнатур атак. Согласно исследованию, 50% обнаруженных уязвимостей в 2009 году повлияли на веб-приложения. В исследовании описывается система, названная HMMPayl, которая анализирует полезную нагрузку HTTP с использованием n-граммов и строит множественные HMM, которые будут использоваться в схеме слияния классификаторов. Описанный мультиклассификатор также можно рассматривать как подход обучения ансамбля; однако, в отличие от ансамблевого метода, Ariu и др. [82] устанавливают классификаторы в конкурентной, а не взаимодополняющей форме. В экспериментальной части, исследование Ariu также использовало набор данных DARPA 1999, а также некоторые другие наборы данных HTTP. В большинстве экспериментов средняя площадь под кривой ROC (Area Under the ROC Curve, AUC) достигается 0,915-0,976 для коэффициента FP, в диапазоне от 10E-4 до 10E-1. Для коэффициентов FP выше 10E-3 HMMPayl достигает степени обнаружения выше чем 0.85, заслуживающий внимания результат. При меньших коэффициентах FP процент обнаруженных атак уменьшается, но по-прежнему остается выше 70% для коэффициента FP 10E-4.

2) Обнаружение аномалий и гибридное обнаружение: HMM были использованы для обнаружения вторжений Joshi и Phoha [83]. Они используют HMM с пятью состояниями и шестью символами наблюдения **для каждого состояния.** Состояния в модели взаимосвязаны таким образом, что любое состояние может быть достигнуто из любого другого состояния. Метод Баума-Велша [84] используется для оценки параметров HMM. Был использован набор данных KDD 1999, и 5 из 41 признака были выбраны для моделирования. PD составлял 79%; Остальные 21% приходятся на коэффициент FP (т.е. классифицирование аномалии как нормального состояния) и коэффициент FN (т.е. классифицирование нормального состояния как атаки). Авторы утверждают, что они смогли значительно повысить точность, используя более пяти признаков. Хотя это является возможным, это далеко не доказано, что, используя больший набор признаков (или полный набор), точность HMM будет увеличиваться.

**I. Индуктивное обучение (Inductive learning)**

Двумя основными методами вывода информации из данных являются дедукция и индукция. Дедукция представляет информацию, которая является логическим следствием информации, представленной в данных, и исходит сверху вниз. Индуктивное рассуждение движется снизу вверх, то есть от конкретных наблюдений к более широким обобщениям и теориям. В индуктивном обучении человек начинает с конкретных наблюдений и измерений, начинает обнаруживать шаблоны и закономерности, формулирует некоторые предварительные гипотезы, которые нужно исследовать, и, наконец, заканчивает разработку некоторых общих выводов или теорий. Некоторые алгоритмы ML являются индуктивными (например, C4.5 для построения деревьев решений), но, когда исследователи ссылаются на индуктивное обучение, обычно они имеют в виду многократное отсечение с приращением для приведения погрешности (n Repeated Incremental Pruning to Produce Error Reduction, RIPPER) [85], а **алгоритм квазиоптимальный** (AQ) [86].

RIPPER строит правила для двухклассовых задач. Он вводит правила непосредственно из обучающих данных, используя подход «Разделяй и властвуй». Он изучает одно правило за раз **таким образом, что правило** охватывает максимальный набор примеров в текущем обучающем наборе. Правило обрезается, чтобы максимизировать желаемый показатель эффективности. Все примеры, которые правильно помечены полученным правилом, исключаются из обучающего набора. Процесс повторяется до тех пор, пока обучающий набор не станет пустым или не будет достигнут предопределенный критерий остановки.

Пример правила в RIPPER [87] может быть:

**Guess: - fail-logins >= 5**

Это означает, что если число неудачных логинов больше или равно 5, то это соединение является «**догадкой» (т.е. предположение парольной атаки)**.

1) Обнаружение злоупотреблений: Lee и др. [87] разработали структуру, в которой использовались несколько методов ML и DM (например, индуктивное обучение, ассоциативные правила, **поиск** **последовательных шаблонов**). Во-первых, **поиск последовательных шаблонов** (также называемое «частотные эпизоды») использовался для построения временных и статистических признаков. **При этом необходимо много экспериментов с временным окном**. Когда поиск последовательных шаблонов выполняется по шаблонам вторжений, он извлекает признаки, которые будут использоваться при обнаружении злоупотреблений. На следующем шаге, эти признаки используются RIPPER для генерации правил. Однако RIPPER не использует все признаки, определенные в своих правилах, поскольку, как и в большинстве алгоритмов классификации, он имеет встроенный механизм отбора признаков.

RIPPER генерирует правила для классификации соединений telnet из набора данных DARPA 1998. Для обучения использовалось **семь недель данных**, и для тестирования использовались **две недели данных**. Тестовые данные содержат 38 типов атак, причем только 14 типов представлены только в тестовых данных. Точность, достигнутая для нового типа атаки, варьировалась от 5.9 (R2L) до 96.7% (Probe или Scan), тогда как для старого типа атаки - от 60 (R2L) до 97% (Probe или Scan). Их метод смог обнаружить с высокой точностью новые атаки, которые являются зондом или сканированием (96,7%) и U2R (81,8%). Точность обнаружения новых типов атак для DoS и R2L была менее 25%.

2) Обнаружение аномалий и гибридное обнаружение. Проблема определения истинной аномалии, по определению, не предлагает аномальный тип данных заранее. Поэтому основная сложность обнаружения аномалий заключается в обнаружении границ между известными и неизвестными категориями [88]. В исследовании был разработан генератор искусственной аномалии, чтобы улучшить эффективность детектора аномалий с точки зрения его способности обобщения. Первым подходом была генерация аномалий на основе распределения, а вторая - **фильтрация** искусственных аномалий. Набор данных DARPA 1998 использовался в **настройке** эксперимента. Достигнута впечатляющая скорость обнаружения аномалий в 94% с 2% FAR. **Обычно желательно**, чтобы FAR был намного меньше 1%, особенно с типами набора данных DARPA 1998, где скорость пакетов на уровне TCP/IP (тысячи пакетов в секунду). Тем не менее, исследование успешно продемонстрировало, как должно проводиться обнаружение истинной аномалии и как данные должны использоваться, **не используя бинарный классификатор для обнаружения аномалий, как и многие другие выполненные исследования.**

**J. Наивный Байес (Naive Bayes)**

Наивные байесовские классификаторы [89] - простые вероятностные классификаторы, применяющие теорему Байеса. Название происходит от того факта, что входные признаки считаются независимыми, тогда как на практике это редко. Условные вероятности p(*C*|*f*1, *f*2, ..., *f*m) формируют модель классификатора, а классификатор присваивает метку класса следующим образом:

где m - число **признаков**, K - число классов, *f*i - i-ый признак, Ck - k-ый класс, p(Ck) - априорная вероятность Ck, а P(*f*i|Ck) - условная вероятность **признака** *f*i данного класса Ck.

Наивные байесовские классификаторы могут обрабатывать произвольное количество независимых признаков независимо от того, являются ли они непрерывными или категоричными. Они сводят задачу оценки большой плотности к оценке плотности одномерного ядра, используя предположение, что признаки независимы.

Хотя наивный байесовский классификатор имеет несколько ограничений, он является оптимальным классификатором, если признаки условно независимы, учитывая истинный класс. Как правило, это один из первых классификаторов, который сравнивается с более сложными алгоритмами. Кроме того, некоторые типы пользователей высказались, что они понимают классификационную модель более интуитивно по сравнению с другими сложными классификаторами (например, SVM). Одним из самых больших преимуществ наивного байесовского классификатора является то, что он является онлайн-алгоритмом, и его обучение может быть завершено за линейное время.

1) Обнаружение злоупотреблений: Panda и Patra [90] использовали наивный байесовский классификатор из пакета Weka [11] и использовали набор данных KDD 1999 для обучения и тестирования. Данные были сгруппированы по четырем типам атак (Probe или Scan, DoS, U2R и R2L), и классификатор достиг 96%, 99%, 90% и 90% точности тестирования по этим категориям, соответственно. **Кумулятивный** FAR составлял 3%.

В документе сравнивались результаты с классификатором нейронной сети и указывалось, что наивный байесовский классификатор имеет более высокую точность, но более высокий FAR, что нежелательно. Результаты, как сообщается, лучше, чем самый высокий **балл**, достигнутый в конкурсе KDD.

2) Обнаружение аномалий и гибридное обнаружение: Фреймворк, разработанный Amor и др. [91] использует простую форму байесовской сети, которая может считаться наивным байесовским классификатором. Построение общей байесовской сети является NP-полной задачей, но простая сеть из корневого узла и листьев для атрибутов формирует наивную байесовскую классификационную структуру.

В этом документе также использовался набор данных KDD 1999 и сгруппированы **по категориям в трех вариантах**, чтобы отразить различные сценарии атак и меры эффективности. Одиночная атака и нормальные данные содержались в первом наборе. Второй набор содержал все четыре типа атак набора данных KDD 1999, и задача, которая должна быть решена, - **это классификация нескольких классов для обнаружения злоупотреблений.** Третий набор содержал нормальные данные и все четыре типа атак (объединенные в одну категорию), и задача, которая должна быть решена, была задачей обнаружения аномалий.

**В документе сообщалось, что результаты достигали точности 97%, 96%, 9%, 12% и 88% для категорий Normal, DoS, R2L, U2R, R2L и Probe или Scan соответственно. В работе не сообщается о ложном срабатывании, но, поскольку нормальная норма 97% достигнута, FAR может считаться менее 3%. Эксперимент по обнаружению аномалий сообщается как 98% и 89% точности для нормальных и ненормальных категорий, соответственно.**

**K. Поиск последовательных шаблонов (Sequential Pattern Mining)**

Поиск последовательных шаблонов появился как один из важных методов DM [92] с появлением транзакционных баз данных, где каждая транзакция имеет временный идентификатор, идентификатор пользователя и набор элементов. Набор элементов представляет собой набор отдельных элементов, приобретенных в транзакции (строгое двоичное представление, в котором элемент был или не был приобретен). Последовательность представляет собой упорядоченный список наборов элементов. Длина последовательности определяется как количество наборов элементов в последовательности. Порядок определяется идентификатором времени. Последовательность A (длины n) содержится во второй последовательности B (длины m), когда все наборы элементов A, {ai} являются подмножествами наборов элементов B, {bj}, с отображением 1-к-1 между индексами i к j таким, что a1 ⊆ bj1, a2 ⊆ bj2, ..., an ⊆ bjk и j1 ≤ j2 ≤ ··· ≤ jk. Другими словами, каждый из наборов элементов A является подмножеством набора элементов в B. Если набор элементов ai является подмножеством набора элементов bj, то следующий набор элементов ai+1 должен быть подмножеством набора элементов bj+m, где m > 0. В последовательности B допускаются наборы элементов, которые не являются подмножеством набора элементов в A (т.е. n ≤ m).

В наборе последовательностей, последовательность A является максимальна, если она не содержится ни в какой другой последовательности. Рассмотрим базу данных *D*, которая содержит последовательности, сгруппированные по определенной именной переменной, такой как IP-адрес, обозначенный *p*. Если последовательность A содержится в *D*(*p*) (т.е. одна из последовательностей *D*(*p*) содержит A), то A поддерживает *D*(*p*). Поддержка - это доля последовательностей в *D*(.), которую поддерживает A. Большая последовательность определяется как последовательность, поддерживающая минимальный порог *Th*. Задача поиска последовательности состоит в том, чтобы найти все максимальные последовательности, которые содержатся в *D* с заданной пользователем минимальной поддержкой *Th*. Максимальные последовательности генерируются сами по себе из последовательностей в *D* путем перечисления всех возможных последовательностей.

1) Обнаружение злоупотреблений: в немного другой области обнаружение вторжений в базы данных [93] использовало поиск последовательных шаблонов для обнаружения вторжений в базы данных путем изучения **шаблонов последовательностей журналов базы данных**. Эти журналы включают в себя несколько полей, таких как тип операции, имя транзакции, идентификатор транзакции, **время начала, время окончания** и т.д. Общий подход, представленный в этой работе, может быть применен к аналогичному анализу вторжений для кибербезопасности, такой как атаки U2R. В исследовании используется алгоритм AprioriAll [92] и генерируется максимальная последовательность с установленным пользователем порогом поддержки. Сгенерированные шаблоны - это сигнатуры для вторжений. Сообщаемая максимальная эффективность составила 91% обнаружения, а наихудший показатель FAR был около 30%. Тем не менее, эта работа успешно применяла поиск последовательных шаблонов для обнаружения вторжений в базы данных путем изучения ее журналов.

2) Обнаружение аномалий и гибридное обнаружение: Li и др. [94] дают хороший пример поиска последовательных шаблонов и его применения для уменьшения избыточности оповещений и минимизации FAR. Они использовали алгоритм AprioriAll для обнаружения многоступенчатых шаблонов атаки. Последовательности были шаблонами атак, ранее обнаруженных или предоставленных кибер-администратором. Для кибер-пользователя также был создан инструмент визуализации шаблонов. Использовался порог поддержки 40%, а генерируемые максимальные последовательности использовались для проверки и корреляции оповещений. В работе использовались данные DARPA 1999 и DARPA 2000 и удалось обнаружить 93% атак за 20 секунд. Во втором наборе экспериментов в исследовании сообщалось о моделированном сценария в реальном времени, где было достигнуто 84% обнаружения атак. В документе говорилось, что порог поддержки был основным параметром для контроля скорости обнаружения и FAR. Необнаруженные шаблоны просто не превышали порог поддержки.

**L. Метод опорных векторов (Support Vector Machine)**

SVM - это классификатор, основанный на нахождении разделительной гиперплоскости в пространстве признаков между двумя классами таким образом, что расстояние между гиперплоскостью и ближайшими точками данных каждого класса максимизируется. Этот подход основан на минимизированном риске классификации [95], а не на оптимальной классификации. SVM хорошо известны своей способностью к обобщению и особенно полезны, когда число признаков, m, велико, а число точек данных, n, мало (m >> n).

Когда два класса не являются разделимыми, добавляются слабые переменные и присваивается параметр стоимости, который назначают для перекрывающихся точек данных. Максимальный запас и место гиперплоскости определяется квадратичной оптимизацией с **практическим** временем выполнения O(n2), размещая SVM среди быстрых алгоритмов, даже когда число атрибутов велико.

Различные типы разделительных поверхностей классификации могут быть реализованы путем применения ядра, такого как линейная, полиномиальная, радиальная базисная функция Гаусса (Radial Basis Function, RBF) или гиперболический тангенс. SVM - это бинарные классификаторы, и многоклассовая классификация реализуется путем разработки SVM для каждой пары классов.

1) Обнаружение злоупотреблений: в работе Li и др. [96] классификатор SVM с ядром RBF использовался для классификации набора данных KDD 1999 в предопределенные категории (DoS, Probe или Scan, U2R, R2L и Normal). Из 41 признака, подмножество атрибутов было выбрано путем следования политике удаления признаков и политике выбора единственного признака. Наконец, в исследовании использовались 19 признаков, определяемых методом выбора признаков.

В той же работе подмножество обучающего набора определялось с помощью оптимизации колонии муравьев (см. Подраздел G). Одним из возможных преимуществ такого подхода является **максимизация обобщения классификатора** и минимизация смещения в наборе KDD, как описано в разделе II. Исследование сообщило о его десятикратной эффективности перекрестной проверки **как общую точность в** 98% с неизвестной дисперсией. Наименьшая производительность в 53% была для категории U2R. Стратегия использования подмножества обучающего набора для преодоления ограничений набора данных **KDD является определенно стоящей,** как показано в этой статье.

Amiri и др. [97] использовали наименьший квадрат SVM, чтобы иметь более быструю систему для обучения на больших наборах данных. Чтобы уменьшить количество признаков в наборе данных KDD с 41 до 19 или менее, они использовали три разных алгоритма выбора признаков. Первый был основан на выборе признака, который максимизирует эффективность классификации, второй была основан на **взаимной** информации, а третий была основан на корреляции. Экспериментальные результаты показали, что подход, основанный на взаимной информации, является более перспективным (хотя и немного), чем два других.

Набор данных был отобран случайным образом, чтобы иметь около 7000 экземпляров (из общего числа 5 миллионов) для каждого из пяти классов (DoS, Probe или Scan, U2R, R2L и Normal). Для повторной установки U2R-атак использовался метод начальной загрузки. Чтобы предсказать тип атаки, для каждой категории были построены пять классификаторов. Таким образом, стоимость связана с каждой категорией и определяется окончательная классификация. Эффективность классификации оценивались как 99% в классах DoS, Probe или Scan, R2L и Normal и 93% в классе U2R с доверительным интервалом 99%.

2) Обнаружение аномалий и гибридное обнаружение: Hu и др. [98] использовали сильный метод опорных векторов (Robust Support Vector Machine, RSVM), разновидность SVM, где **разделяющая** гиперплоскость усредняется, чтобы быть более **гладкой**, а параметр регуляризации определяется автоматически в качестве классификатора аномалий в их исследовании. Части базового модуля безопасности [18] из набора данных DARPA 1998 использовалась, чтобы предварительно обработать обучающие и тестовые данные. Исследование показало хорошую эффективность классификации при наличии шума (такого как некорректная маркировка обучающего набора данных) и сообщило о точности в 75% без ложных тревог и 100% точности с 3% FAR.

Wagner и др. [99] использовали данные NetFlow, собранные из **реального мира**, и смоделированные данные об атаках, используя инструмент Flame [100] и другие источники **интернет-провайдера**, который предоставил данные об атаках **в реальном мире**, такие как сканирование NetBIOS, атаки DoS, спам POP и сканирование Secure Shell (SSH). В исследовании использовался одноклассовый классификатор SVM, который считается естественным подходом для обнаружения аномалий. Было введено **новое оконное ядро**, чтобы помочь найти аномалию, основанную на временной позиции данных NetFlow. Более чем одна записть NetFlow вошла в это ядро, предполагая, что последовательная информация была также сохранена для классификации. Кроме того, окно было проверено на IP-адрес и объем трафика. Оконное ядро работает как мера подобия между аналогичными последовательными записями NetFlow. Эффективность была указана с точностью от 89% до 94% при различных типах атак с коэффициентом FP от 0% до 3%.

Подход, описанный Shon и Moon [101], является **основой** для обнаружения новых атак в сетевом трафике. Их подход представляет собой комбинацию самоорганизующейся карты признаков (Self-Organizing Feature Map, SOFM), GA и SVM. Это описано здесь, потому что основной новизной метода является усиленный SVM.

Четыре этапа в предлагаемой системе следующие:

* Тип неконтролируемой ИНС, выполняющей кластеризацию SOFM [102], используется для профилирования пакетов.
* Фильтрация пакетов выполняется с использованием пассивных **слепков** TCP/IP.
* GA выбирает признаки.
* Данные классифицируются с использованием **усиленного** SVM, который получен из одноклассового SVM и контролируемого SVM с мягким **пределом**. Первый предоставляет возможность немеченой классификации одноклассового SVM, а второй обеспечивает высокую эффективность обнаружения контролируемого SVM.

В исследовании использовался набор данных обнаружения вторжений DARPA 1999. Чтобы сделать набор данных более реалистичным, выбранное подмножество состояло от 1% до 1.5% атак и от 98.5% до 99% обычного трафика. Результаты **усиленного** SVM имели точность 87.74%, коэффициент FP в 10% и коэффициент FN в 27.27%. Эти результаты были значительно лучше, чем из одноклассовых SVM, но не так хороши, как SVM с мягким **пределом**. Однако, **усиленный** SVM может обнаруживать новые шаблоны атак, тогда как SVM с мягким **пределом** не может.

Другой эксперимент был выполнен на наборе данных, собранном в институте, где работает один из авторов. Эффективность системы, основанной на **усиленном** SVM была сравнима с эффективность Snort [16] и Bro [103]. Тестирование проводилось на нормальным данных, двух наборах известных атак и так называемых реальных данных (т.е. данные, собранным институтом автора). В случае нормальных данных (см. Таблицу VI), Bro выполнил лучше. Snort и **усиленный** SVM выполнялись аналогично, причем Snort был немного лучше. В случае известных атак, эффективность Snort и Bro была лучше, чем эффективность **усиленного** SVM. В случае реальных данных, **усиленный** SVM превосходит Snort и Bro.

**V. Вычислительная сложность методов машинного обучения и анализа данных**

**(V. COMPUTATIONAL COMPLEXITY OF ML AND DM**

**METHODS)**

В литературе содержится ограниченное количество сравнений производительности для алгоритмов ML и DM. Концепция калибровки предсказания **предполагает** своего рода сглаживание **на выходе** предсказаний, чтобы они соответствовали их распределению. Поэтому соответствующее сравнение эффективности должно как не включать, так и включать калибровку предсказаний с подходящим подходом, таким как шкалирование по Платту и изотонная регрессия [104]. Согласно эмпирическому сопоставлению в [104], мешковые деревья, случайные леса и ИНС дают наилучшие результаты. После калибровки улучшенные деревья и SVM работают лучше. В исследовании также сообщается, что обобщения не выполняются, существует значительная изменчивость в отношении задач и показателей, а характеристики модели не всегда согласованы. Хотя некоторые алгоритмы считаются более эффективными, чем другие, производительность конкретного алгоритма ML зависит от применения и реализации.

В таблице VII представлена ​​вычислительная сложность (т.е. временная сложность) различных алгоритмов ML и DM. Элементы таблицы VII были найдены посредством обширной литературы и поиска в Интернете. Естественно, некоторые временные сложности являются спорными, и они основаны на опыте пользователя и навыках исполнителя. Большинство из этих алгоритмов имеют хорошо поддерживаемые реализации с открытым исходным кодом. Предположения, сделанные в таблице VII, о том, что данные состоят из *n* экземпляров, каждый из которых описан *m* атрибутами, а *n* намного больше *m*.

Как правило, алгоритмы O(n) и O(n log n) считаются линейными и могут использоваться для онлайн-подходов. O(n2) считается приемлемой временной сложностью для большинства практик. O(n3) и выше считаются значительно более медленными алгоритмами и используются для оффлайн подходов.

Более высокий процент работ, рассматриваемых в этом обзоре, представляют их подходы как оффлайн методы. Обработанные данные готовы и введены в систему в целом. Когда системный конвейер предназначен для работы в качестве онлайн системы или, когда система обрабатывает потоковые данные, необходимо решить несколько проблем, таких как потоки ввода/вывода данных и буферизация, запуск методов онлайн и отображение результатов с помощью подходящей временной информации. В нескольких исследованиях [35], [42], [52], [62] описаны их системы как работающие онлайн и обрабатывающие входные данные в мягком режиме реального времени. Интересно, что в некоторых из этих исследований даже используются более медленные алгоритмы, такие как анализ последовательностей [52] для выполнения обнаружения вторжений. Также существуют проблемы системного уровня, такие как разбиение входных потоков данных (например, framework MapReduce [105]), использование методов обучения и сбор и объединение результатов параллельно.

В общем, при подготовке моделей прогнозирования или изучении признаков сетевого трафика, подходящий онлайновый метод обращается, как минимум, к трем факторам: временная сложность, возможность дополнительного обновления и возможность обобщения.

* Временная сложность каждого алгоритма представлена ​​в таблице VII. Метод должен быть близок к приблизительно O(n log n), чтобы считаться потоковым алгоритмом. Однако медленные алгоритмы, такие как методы анализа последовательностей или ИНС, также используются в потоковых системах, сохраняя входные данные в окне и имеющие небольшой n.
* Для возможности постепенного обновления, алгоритмы кластеризации, статистические методы (например, HMM, байесовские сети) и ансамблевые модели могут быть легко обновлены постепенно [89], [106]. Тем не менее, обновления для ИНС, SVM или эволюционных моделей могут вызывать осложнения [89], [106].
* Требуется хорошая обобщающая способность, так чтобы обученная модель кардинально не отклонялась от исходной модели, когда **видны** новые входные данные. Большинство современных методов ML и DM имеют очень хорошую обобщающую способность.

Этап тестирования для методов обычно выполняется быстро, в основном, **за линейное время** по отношению к размеру входных данных. Поэтому, **однажды** обученные, большинство методов можно использовать онлайн.

**VI. Замечания и рекомендации**

**(VI. OBSERVATIONS AND RECOMMENDATIONS)**

Объем работ, **найденных на** ML и DM для обнаружения кибер-вторжений, показывает, что эти методы являются распространенной и растущей областью исследований для кибербезопасности. Возникает вопрос: какой из этих методов наиболее эффективный для кибер-приложений? К сожалению, это еще не установлено.

**А. Замечания, относящиеся к наборам данных (Observations Related to the Data Sets)**

В таблице VIII представленные документы методов ML и DM применялись к **кибер** области, которые были рассмотрены (и описаны в Разделе IV), включающие количество раз, которое они были процитированы, **кибер** задачу, которую они решают, и используемые данные. Интересно, что из 39 работ, перечисленных в таблице VIII, 28, использовали данные DARPA 1998, DARPA 1999, DARPA 2000 или KDD 1999. Только две использовали наборы данных NetFlow, две использовали данные tcpdump, одна использовала данные DNS, одна использовала SSH-команды, а четыре использовали некоторые другие типы данных. При определении представленных документов, это исследование, в основном, рассматривало ML или DM методы, которые использовали авторы, и тот факт, что эти работы представляли собой **подходы для обнаружения злоупотреблений, аномалий или гибридный подход**. Другой важный фактор заключался в том, что работы были **высоко упоминаемыми, что считалось показателями их качества.** Тем не менее, некоторые перспективные появившиеся методы также были включены, **даже если они еще не имели шанса быть высоко упоминаемыми**. Хотя исследование было нацелено на работы, написанные в 2000 году или позже, два предыдущих документа были хорошо написаны и высоко цитировались и, следовательно, заслуживали включения в эту обзорную статью.

Тот факт, что многие работы используют наборы данных DARPA и KDD, связан с тем, насколько сложным и трудоемким является получение представленного набора данных. Как только такой набор данных доступен, исследователи склонны повторно его использовать. Кроме того, повторное использование одного и того же набора данных должно позволять легко сравнивать точность различных методов. Как обсуждалось ранее, в случае наборов данных DARPA и KDD это было фактически не совсем так, потому что эти наборы данных настолько велики, что исследователи решают работать с разными подмножествами. В двух работах, в которых обсуждалось использование NetFlow, также обсуждалась эффективность обнаружения аномалий. Это очевидно, потому что NetFlow не обладает таким богатым набором признаков, как tcpdump или DARPA, или KDD, а также признаками, необходимыми для обнаружения определенных сигнатур при обнаружении злоупотреблений. (Его признаки ограничены информацией о потоках, генерируемой **маршрутизаторами более высокого класса**.)

**В. Факторы, связанные с производительностью IDS (Factors Related to IDS Performance)**

Одним из наиболее важных факторов, влияющих на производительность IDS, является тип и уровень входных данных. Как обсуждалось ранее, в нескольких исследованиях использовались наборы данных DARPA или KDD, поскольку их легко получить, и они содержат данные сетевого уровня (либо tcpdump, либо NetFlow), а также данные уровня ОС (например, сетевые журналы, журналы безопасности, системные вызовы ядра). В первую очередь, данные атак, поступающие в сетевой стек, и влияние этих пакетов на уровень ОС несли важную информацию. Как результат, предпочтительно, чтобы IDS была способна получать данные сети и ядра. Если для IDS доступны только данные NetFlow (гораздо проще получить и обработать), эти данные должны быть дополнены данными сетевого уровня, такими как сетевые датчики, которые генерируют дополнительные признаки пакетов или потоков. Если возможно, сетевые данные должны быть дополнены данными уровня ядра ОС. Как было обнаружено, несколько исследований приблизились к задаче обнаружения вторжений путем изучения команд уровня ОС (то есть IDS на основе хоста), а не сетевых пакетов.

Второй фактор, связанный с производительностью IDS, - это тип используемых алгоритмов ML и DM и общий дизайн системы. Обзор литературы показал, что во многих исследованиях использовались наборы данных DARPA и KDD и применялись различные методы ML. Эти исследования на самом деле не построили IDS, но изучали характеристики методов ML и DM для некоторых данных кибербезопасности. Тем не менее, классификация исследований в отношении членства авторов показывает исследования, в которых строились фактические IDS, и использовались данные реального мира, полученные из сетей кампуса или магистральных сетей Интернета. Все эти исследования, как представляется, использовали системы, интегрированные с более чем одним методом ML, и несколько модулей, связанных с захватом сигнатур, сигнатурной базой данных и т. Д.

**C. Критерии сравнения (Comparison Criteria)**

Существует несколько критериев, по которым можно сравнить методы ML / DM для кибер:

* Точность
* Время для обучения модели
* Время для классификации неизвестного экземпляра с обученной моделью
* Понятность окончательного решения (классификации)

Если сравнить точность нескольких методов ML / DM, эти методы должны быть обучены точно таким же данным тренинга и проверены точно на одних и тех же данных тестирования. К сожалению, даже в тех исследованиях, которые использовали один и тот же набор данных (например, KDD 1999), когда они сравнивали свои результаты с лучшими методами из Кубка KDD (и обычно утверждали, что их результаты были лучше), они делали это в несовершенной форме - Они использовали подмножество набора данных KDD, но не обязательно одно и то же подмножество, которое использовал другой метод. Поэтому точность этих результатов не сопоставима.

Время для обучения модели является важным фактором из-за постоянно меняющихся типов и особенностей кибер-атаки. Даже аномальные детекторы должны часто обучаться, возможно, постепенно, с новыми обновлениями сигнатур вредоносных программ.

Время для классификации нового экземпляра является важным фактором, который отражает время реакции и мощность обработки пакетов системы обнаружения вторжений.

Понятность или читаемость модели классификации - это средство, помогающее администраторам легко изучить возможности модели, чтобы быстрее исправлять свои системы. Эта информация (такая как тип пакета, номер порта или какая-либо другая функция сетевого пакета высокого уровня, которая отражает троп кибер-атаки) будет доступна через векторы признаков, которые помечаются классификатором как категория вторжения.

**D. Особенности машинного обучения и анализа данных для кибер (Peculiarities of ML and DM for Cyber)**

ML и DM были чрезвычайно полезны во многих приложениях. В кибер-домене есть некоторые особенности, которые затрудняют использование этих методов. Эти особенности особенно связаны с тем, как часто требуется переучивание модели и наличие помеченных данных.

В большинстве приложений ML и DM модель (например, классификатор) обучается, а затем используется в течение длительного времени без каких-либо изменений. В этих приложениях предполагается, что процессы являются квазистационарными, и поэтому переобучение модели происходит не часто. Ситуация с обнаружением кибер-вторжений различна. Модели ежедневно тренируются [56], когда аналитик требует [43], или каждый раз, когда идентифицируется новое вторжение, и его образ становится известным [75]. Особенно, когда модели должны ежедневно обучаться, их время обучения становится важным. (Это, безусловно, займет менее одного полного дня, чтобы переучивать модель.) Традиционно методы ML и DM начинают обучение с нуля. Однако, если часто требуется переучивать модель (например, ежедневно) из-за всего лишь нескольких изменений в данных, имеет смысл начать с обученной модели и продолжить ее обучение или использовать самоадаптивные модели. Плодотворная область исследований будет заключаться в исследовании методов быстрого инкрементного обучения, которые могут быть использованы для ежедневных обновлений моделей для обнаружения неправильного использования и аномалии.

Существует много доменов, в которых легко получить учебные данные, и в этих областях обычно развиваются методы ML и DM (например, рекомендации, которые Amazon делает для своих клиентов). В других областях, где трудно получить данные (например, данные мониторинга здоровья для машин или самолетов), применение ML и DM может быть в изобилии. В кибер-домене многие данные могут быть собраны путем размещения датчиков в сетях (например, для получения NetFlow или TCP), что, хотя и не простая задача, безусловно, стоит того.

Однако есть проблема с огромным объемом этих данных - слишком много данных для хранения (терабайт в день). Другая проблема заключается в том, что некоторые данные должны быть помечены как полезные, что может быть трудоемкой задачей. Данные для обучения определенно должны быть помечены, и это даже верно для методов обнаружения чистой аномалии. Эти методы должны использовать данные, которые являются нормальными; Они не могут разработать модель с смежными данными атаки. Кроме того, поскольку они должны быть протестированы с новыми атаками, требуются также некоторые новые данные атаки. Наибольший разрыв наблюдается в доступности помеченных данных, и, безусловно, стоит инвестировать в сбор данных и маркировку некоторых из них. Используя этот новый набор данных, можно было бы добиться значительных успехов в методах ML и DM в кибербезопасности и могли бы стать прорывами. В противном случае наилучшим доступным набором данных прямо сейчас является скорректированный набор данных KDD 1999. (Однако, будучи 15 лет, этот набор данных не содержит примеров всех новых атак, которые произошли за последние 15 лет.)

**Е. Рекомендации ML и DM для обнаружения неправомерного использования и обнаружения аномалий (ML and DM Recommendations for Misuse and Anomaly Detection)**

IDS обычно гибридны и имеют модули обнаружения аномалий и обнаружения злоупотреблений. Модуль обнаружения аномалий классифицирует ненормальный сетевой трафик. Модуль обнаружения злоупотреблений классифицирует шаблоны атаки с известными сигнатурами или извлекает новые подписи из данных, помеченных атакой, поступающих из модуля аномалий.

Часто детектор аномалий основан на методе кластеризации. Среди алгоритмов кластеризации методы на основе плотности (например, DBSCAN) являются самыми универсальными, легкими в реализации, менее зависимыми от параметра или распределения и имеют высокую скорость обработки. В аномальных детекторах SVM с одним классом также хорошо работают, и многое можно узнать, извлекая правила ассоциации или последовательные шаблоны из доступных обычных данных трафика.

Среди детекторов неправильного использования, поскольку необходимо подписать подписи, важно, чтобы классификатор мог генерировать читаемые сигнатуры, такие как особенности ветвления в дереве решений, гены в генетическом алгоритме, правила в правилах правил ассоциации или последовательности в последовательности Добыча. Поэтому классификаторы черного ящика, такие как ANN и SVM, плохо подходят для обнаружения неправильного использования.

Несколько самых современных алгоритмов ML и DM подходят для обнаружения неправильного использования. Некоторые из этих методов являются статистическими, такими как байесовские сети и HMM; Некоторые из них являются энтропированными, такими как деревья решений; Некоторые из них являются эволюционными, такими как генетические алгоритмы; Некоторые из них - ансамблевые методы, такие как Random Forests; И некоторые из них основаны на правилах ассоциации. Разработчики системы должны исследовать, имеют ли данные обучения достаточно хорошее качество и имеют статистические свойства, которые могут быть использованы (например, распределение Гаусса). Также важно знать, будет ли требуемая система работать в сети или в автономном режиме. Ответы на такие вопросы будут определять наиболее подходящий подход ML. По мнению авторов этой статьи, сетевые данные не могут быть правильно смоделированы с использованием простого распределения (например, гауссовского) из-за того, что на практике один сетевой пакет может содержать полезную нагрузку, которая может быть связана с десятками Сетевых протоколов и поведения пользователей [113]. Изменчивость полезной нагрузки характеризуется суммой множественных вероятностных распределений или совместных распределений вероятностей, которые не являются непосредственно разделяемыми. Поэтому методы, подобные байесовским сетям или HMM, могут быть не самым сильным подходом, потому что данные не обладают свойствами, которые наиболее подходят для них. Эволюционные методы расчета могут занять много времени и, следовательно, могут не подходить для систем, которые работают в режиме онлайн. Если данные обучения недостаточны, у Random Forest может быть преимущество. Если захват сигнатуры атаки важен, деревья решений, эволюционные вычисления и правила ассоциации могут быть полезны.

**VII. Выводы**

**(CONCLUSIONS)**

В **статье описывается обзор литературы по методам ML и DM, используемым для кибербезопаснсти.** Особый акцент был сделан на поиске примеров работ, в которых описывается использование различных методов ML и DM в области киберугроз, как для обнаружения злоупотреблений, так и аномалий. К сожалению, не установлены методы, наиболее эффективные для кибер-приложений; и, учитывая богатство и сложность методов, невозможно составить одну рекомендацию для каждого метода в зависимости от типа атаки, которую система должна обнаружить. При определении эффективности методов существует не один, а несколько критериев, которые необходимо учитывать. Они включают (как описано в разделе VI, подраздел C) точность, сложность, время для классификации неизвестного экземпляра с обученной моделью и понятность окончательного решения (классификации) каждого метода ML или DM. В зависимости от конкретной IDS, некоторые из них могут быть важнее других.

Другим важным аспектом ML и DM для обнаружения кибер-вторжений является важность наборов данных для обучения и тестирования систем. Методы ML и DM не могут работать без репрезентативных данных, и для таких наборов данных трудно и трудоемко. Для возможности обнаружения аномалий и обнаружения неправильного использования, для IDS выгодно иметь доступ к данным на уровне сети и ядра. Если доступны только данные NetFlow, эти данные должны быть дополнены данными сетевого уровня, такими как сетевые датчики, которые генерируют дополнительные функции пакетов или потоков. Если возможно, сетевые данные должны быть дополнены данными уровня ядра ОС.

Наибольший разрыв наблюдается в доступности помеченных данных, и, безусловно, стоит инвестировать в сбор данных и их маркировку. Используя этот новый набор данных, несколько перспективных методов ML и DM можно было бы использовать для разработки моделей и сравнения, сужая список ML и DM, эффективных для кибер-приложений. Значительные успехи могут быть достигнуты для методов ML и DM в кибербезопасности с использованием этого набора данных, и прорывы могут быть возможны.

Существуют некоторые особенности кибер-проблемы, которые затрудняют использование методов ML и DM (как описано в разделе VI, подраздел D). Они особенно связаны с тем, как часто нужно переучивать модель. Плодотворная область исследований будет заключаться в исследовании методов быстрого **дополнительного** обучения, которые могут быть использованы для ежедневных обновлений моделей для обнаружения злоупотреблений и аномалий.

REFERENCES

[1] A. Mukkamala, A. Sung, and A. Abraham, “Cyber security challenges: Designing efficient intrusion detection systems and antivirus tools,” in Enhancing Computer Security with Smart Technology, V. R. Vemuri, Ed. New York, NY, USA: Auerbach, 2005, pp. 125–163.

[2] M. Bhuyan, D. Bhattacharyya, and J. Kalita, “Network anomaly detection: Methods, systems and tools,” IEEE Commun. Surv. Tuts., vol. 16, no. 1, pp. 303–336, First Quart. 2014.

[3] T. T. T. Nguyen and G. Armitage, “A survey of techniques for internet traffic classification using machine learning,” IEEE Commun. Surv. Tuts., vol. 10, no. 4, pp. 56–76, Fourth Quart. 2008.

[4] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, “Anomaly-based network intrusion detection: Techniques, systems and challenges,” Comput. Secur., vol. 28, no. 1, pp. 18–28, 2009.

[5] A. Sperotto, G. Schaffrath, R. Sadre, C. Morariu, A. Pras, and B. Stiller, “An overview of IP flow-based intrusion detection,” IEEE Commun. Surv. Tuts., vol. 12, no. 3, pp. 343–356, Third Quart. 2010.

[6] S. X. Wu and W. Banzhaf, “The use of computational intelligence in intrusion detection systems: A review,” Appl. Soft Comput., vol. 10, no. 1, pp. 1–35, 2010.

[7] Y. Zhang, L. Wenke, and Y.-A. Huang, “Intrusion detection techniques for mobile wireless networks,” Wireless Netw., vol. 9, no. 5, pp. 545–556, 2003.

[8] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, “The KDD process for extracting useful knowledge from volumes of data,” Commun. ACM, vol. 39, no. 11, pp. 27–34, 1996.

[9] C. Shearer, “The CRISP-DM model: The new blueprint for data mining,”

J. Data Warehouse., vol. 5, pp. 13–22, 2000.

[10] A. Guazzelli, M. Zeller, W. Chen, and G. Williams, “PMML an open standard for sharing models,” R J., vol. 1, no. 1, pp. 60–65, May 2009.

[11] M. Hall, E. Frank, J. Holmes, B. Pfahringer, P. Reutemann, and I. Witten, “The WEKA data mining software: An update,” ACM SIGKDD Explor. Newslett., vol. 11, no. 1, pp. 10–18, 2009.

[12] R Language Definition. (2000). R Core Team [Online]. Available: <ftp://155.232.191.133/cran/doc/manuals/r-devel/R-lang.pdf>, accessed on Nov. 2015.

[13] M. Graczyk, T. Lasota, and B. Trawinski, “Comparative analysis of premises valuation models using KEEL, RapidMiner, and WEKA,” Computational Collective Intelligence. Semantic Web, Social Networks and Multiagent Systems. New York, NY, USA: Springer, 2009, pp. 800–812.

[14] V. Jacobson, C. Leres, and S. McCanne, The Tcpdump Manual Page. Berkeley, CA, USA: Lawrence Berkeley Laboratory, 1989.

[15] G. Combs. Wireshark [Online]. Available: [http://www.wireshark.org](http://www.wireshark.org/), accessed on Jun. 2014.

[16] Snort 2.0. Sourcefire [Online]. Available: <http://www.sourcefire.com/technology/whitepapers.htm>, accessed on Jun. 2014.

[17] G. F. Lyon, Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning. USA: Insecure, 2009.

[18] R. Lippmann, J. Haines, D. Fried, J. Korba, and K. Das, “The 1999 DARPA offline intrusion detection evaluation,” Comput. Netw., vol. 34, pp. 579–595, 2000.

[19] R. Lippmann et al., “Evaluating intrusion detection systems: The 1998 DARPA offline intrusion detection evaluation,” in Proc. IEEE DARPA Inf. Surviv. Conf. Expo., 2000, pp. 12–26.

[20] S. J. Stolfo, KDD Cup 1999 Data Set, University of California Irvine, KDD repository [Online]. Available: [http://kdd.ics.uci.edu](http://kdd.ics.uci.edu/), accessed on Jun. 2014.

[21] M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, “A detailed analysis of the KDD Cup 1999 data set,” in Proc. 2nd IEEE Symp. Comput. Intell. Secur. Defense Appl., 2009, pp. 1–6.

[22] K. Hornik, M. Stinchcombe, and H. White, “Multilayer feedforward networks are universal approximators,” Neural Netw., vol. 2, pp. 359–366, 1989.

[23] F. Rosenblatt, “The perceptron: A probabilistic model for information storage and organization in the brain,” Psychol. Rev., vol. 65, no. 6, pp. 386–408, 1958.

[24] J. Cannady, “Artificial neural networks for misuse detection,” in Proc. 1998 Nat. Inf. Syst. Secur. Conf., Arlington, VA, USA, 1998, pp. 443–456.

[25] Internet Security Scanner (ISS). IBM [Online]. Available: [http://www.iss.net](http://www.iss.net/), accessed on Feb. 2015.

[26] B. Morel, “Artificial intelligence and the future of cybersecurity,” in Proc. 4th ACM Workshop Secur. Artif. Intell., 2011. pp. 93–98.

[27] R. P. Lippmann and R. K. Cunningham, “Improving intrusion detection performance using keyword selection and neural networks,” Comput. Netw., vol. 34, pp. 597–603, 2000.

[28] A. Bivens, C. Palagiri, R. Smith, B. Szymanski, and M. Embrechts, “Network-based intrusion detection using neural networks,” Intell. Eng. Syst. Artif. Neural Netw., vol. 12, no. 1, pp. 579–584, 2002.

[29] R. Agrawal, T. Imielinski, and A. Swami, “Mining association rules between sets of items in large databases,” in Proc. Int. Conf. Manage. Data Assoc. Comput. Mach. (ACM), 1993, pp. 207–216.

[30] C. M. Kuok, A. Fu, and M. H. Wong, “Mining fuzzy association rules in databases,” ACM SIGMOD Rec., vol. 27, no. 1, pp. 41–46, 1998.

[31] L. Zadeh, “Fuzzy sets,” Inf. Control, vol. 8, no. 3, pp. 338—35, 1965.

[32] H. Brahmi, B. Imen, and B. Sadok, “OMC-IDS: At the cross-roads of OLAP mining and intrusion detection,” in Advances in Knowledge Discovery and Data Mining. New York, NY, USA: Springer, 2012, pp. 13–24.

[33] H. Zhengbing, L. Zhitang, and W. Junqi, “A novel network intrusion detection system (NIDS) based on signatures search of data mining,” in Proc. 1st Int. Conf. Forensic Appl. Techn. Telecommun. Inf. Multimedia Workshop (e-Forensics ‘08), 2008, pp. 10–16.

[34] H. Han, X. Lu, and L. Ren, “Using data mining to discover signatures in network-based intrusion detection,” in Proc. IEEE Comput. Graph. Appl., 2002, pp. 212–217.

[35] D. Apiletti, E. Baralis, T. Cerquitelli, and V. D’Elia, “Characterizing network traffic by means of the NetMine framework,” Comput. Netw., vol. 53, no. 6, pp. 774–789, Apr. 2009.

[36] NetGroup, Politecnico di Torino, Analyzer 3.0 [Online]. Available: [http://analyzer.polito.it](http://analyzer.polito.it/), accessed on Jun. 2014.

[37] E. Baralis, T. Cerquitelli, and V. D’Elia. (2008). Generalized Itemset Discovery by Means of Opportunistic Aggregation. Tech. Rep., Politecnico di Torino [Online] <https://dbdmg.polito.it/twiki/bin/view/Public/NetworkTrafficAnalysis>, accessed on Jun. 2014.

[38] A. Tajbakhsh, M. Rahmati, and A. Mirzaei, “Intrusion detection using fuzzy association rules,” Appl. Soft Comput., vol. 9, pp. 462–469, 2009.

[39] J. Luo and S. Bridges, “Mining fuzzy association rules and fuzzy frequency episodes for intrusion detection,” Int. J. Intell. Syst., vol. 15, no. 8, pp. 687–703, 2000.

[40] D. Heckerman, A Tutorial on Learning with Bayesian Networks. New York, NY, USA: Springer, 1998.

[41] F. V. Jensen, Bayesian Networks and Decision Graphs. New York, NY, USA: Springer, 2001.

[42] C. Livadas, R. Walsh, D. Lapsley, and W. Strayer, “Using machine learning techniques to identify botnet traffic,” in Proc 31st IEEE Conf. Local Comput. Netw., 2006, pp. 967–974.

[43] F. Jemili, M. Zaghdoud, and A. Ben, “A framework for an adaptive intrusion detection system using Bayesian network,” in Proc. IEEE Intell. Secur. Informat., 2007, pp. 66–70.

[44] C. Kruegel, D. Mutz, W. Robertson, and F. Valeur, “Bayesian event classification for intrusion detection,” in Proc. IEEE 19th Annu. Comput. Secur. Appl. Conf., 2003, pp. 14–23.

[45] S. Benferhat, T. Kenaza, and A. Mokhtari, “A Naïve Bayes approach for detecting coordinated attacks,” in Proc. 32nd Annu. IEEE Int. Comput. Software Appl. Conf., 2008, pp. 704–709.

[46] K. Jain and R. C. Dubes, Algorithms for Clustering Data. Englewood Cliffs, NJ, USA: Prentice-Hall, 1988.

[47] K. Leung and C. Leckie, “Unsupervised anomaly detection in network intrusion detection using clusters,” in Proc. 28th Australas. Conf. Comput. Sci., vol. 38, 2005, pp. 333–342.

[48] P. W. Holland and S. Leinhardt, “Transitivity in structural models of small groups,” Comp. Group Stud., vol. 2, pp. 107–124, 1971.

[49] J. Watts and S. Strogatz, “Collective dynamics of ‘small-world’ networks,”

Nature, vol. 393, pp. 440–442, Jun. 1998.

[50] R. Hendry and S. J. Yang, “Intrusion signature creation via clustering anomalies,” in Proc. SPIE Defense Secur. Symp. Int. Soc. Opt. Photonics, 2008, pp. 69730C–69730C.

[51] M. Blowers and J. Williams, “Machine learning applied to cyber operations,” in Network Science and Cybersecurity. New York, NY, USA: Springer, 2014, pp. 55–175.

[52] K. Sequeira and M. Zaki, “ADMIT: Anomaly-based data mining for intrusions,” in Proc 8th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., 2002, pp. 386–395.

[53] R. Quinlan, “Induction of decision trees,” Mach. Learn., vol. 1, no. 1, pp. 81–106, 1986.

[54] R. Quinlan, C4.5: Programs for Machine Learning. San Mateo, CA, USA: Morgan Kaufmann, 1993.

[55] C. Kruegel and T. Toth, “Using decision trees to improve signaturebased intrusion detection,” in Proc. 6th Int. Workshop Recent Adv. Intrusion Detect., West Lafayette, IN, USA, 2003, pp. 173–191.

[56] L. Bilge, E. Kirda, C. Kruegel, and M. Balduzzi, “EXPOSURE: Finding malicious domains using passive DNS analysis,” presented at the 18th Annu. Netw. Distrib. Syst. Secur. Conf., 2011.

[57] L. Bilge, S. Sen, D. Balzarotti, E. Kirda, and C. Kruegel, “2014 Exposure: A passive DNS analysis service to detect and report malicious domains,” ACM Trans. Inf. Syst. Secur., vol. 16, no. 4, Apr. 2014.

[58] R. Polikar, “Ensemble based systems in decision making,” IEEE Circuits Syst. Mag., vol. 6, no. 3, pp. 21–45, Third Quart. 2006.

[59] Y. Freund and R. Schapire, “Experiments with a new boosting algorithm,” in Proc. 13th Int. Conf. Mach. Learn., 1996, vol. 96, pp. 148–156.

[60] P. Long and R. Servedio, “Boosting the area under the ROC curve,” Adv. Neural Inf. Process. Syst., pp. 945–952, 2007.

[61] L. Breiman, “Random forests,” Mach. Learn., vol. 45, no. 1, pp. 5–32, 2001.

[62] J. Zhang, M. Zulkernine, and A. Haque, “Random-forests-based network intrusion detection systems,” IEEE Trans. Syst. Man Cybern. C: Appl. Rev., vol. 38, no. 5, pp. 649–659, Sep. 2008.

[63] F. Gharibian and A. Ghorbani, “Comparative study of supervised machine learning techniques for intrusion detection,” in Proc. 5th Annu. Conf. Commun. Netw. Serv. Res., 2007, pp. 350–358.

[64] J. H. Friedman, “Multivariate adaptive regression splines,” Anal. Statist., vol. 19, pp. 1–141, 1991.

[65] S. Mukkamala, A. Sunga, and A. Abraham, “Intrusion detection using an ensemble of intelligent paradigms,” J. Netw. Comput. Appl., vol. 28, no. 2, pp. 167–182, 2004.

[66] L. Bilge, D. Balzarotti, W. Robertson, E. Kirda, and C. Kruegel, “Disclosure: Detecting botnet command and control servers through large-scale netflow analysis,” in Proc. 28th Annu. Comput. Secur. Appl. Conf. (ACSAC’12), Orlando, FL, USA, Dec. 3–7, 2012, pp. 129–138.

[67] D. E. Goldberg and J. H. Holland, “Genetic algorithms and machine learning,” Mach. Learn., vol. 3, no. 2, pp. 95–99, 1988.

[68] J. R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection. Cambridge, MA, USA: MIT Press, 1992.

[69] H. G. Beyer and H. P. Schwefel, “Evolution strategies: A comprehensive introduction,” J. Nat. Comput., vol. 1, no. 1, pp. 3–52, 2002.

[70] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in Proc. IEEE Int. Conf. Neural Netw., 1995, vol. IV, pp. 1942–1948.

[71] M. Dorigo and L. M. Gambardella, “Ant colony system: A cooperative learning approach to the traveling salesman problem,” IEEE Trans. Evol. Comput., vol. 1, no. 1, pp. 53–66, Apr. 1997.

[72] J. Farmer, N. Packard, and A. Perelson, “The immune system, adaptation and machine learning,” Phys. D: Nonlinear Phenom., vol. 2, pp. 187–204, 1986.

[73] W. Li, “Using genetic algorithms for network intrusion detection,” in Proc. U.S. Dept. Energy Cyber Secur. Group 2004 Train. Conf., 2004, pp. 1–8.

[74] A. Abraham, C. Grosan, and C. Martin-Vide, “Evolutionary design of intrusion detection programs,” Int. J. Netw. Secur., vol. 4, no. 3, pp. 328–339, 2007.

[75] J. Hansen, P. Lowry, D. Meservy, and D. McDonald, “Genetic programming for prevention of cyberterrorism through dynamic and evolving intrusion detection,” Decis. Support Syst., vol. 43, no. 4, pp. 1362–1374, Aug. 2007.

[76] S. Khan, “Rule-based network intrusion detection using genetic algorithms,” Int. J. Comput. Appl., vol. 18, no. 8, pp. 26–29, Mar. 2011.

[77] T. Jolliffe, Principal Component Analysis, 2nd ed. New York, NY, USA: Springer, 2002.

[78] W. Lu and I. Traore, “Detecting new forms of network intrusion using genetic programming,” Comput. Intell., vol. 20, pp. 470–489, 2004.

[79] A. Markov, “Extension of the limit theorems of probability theory to a sum of variables connected in a chain,” Dynamic Probabilistic Systems, vol. 1, R. Howard. Hoboken, NJ, USA: Wiley, 1971 (Reprinted in Appendix B).

[80] L. E. Baum and J. A. Eagon, “An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology,” Bull. Amer. Math. Soc., vol. 73, no. 3, p. 360, 1967.

[81] A. Arnes, F. Valeur, G. Vigna, and R. A. Kemmerer, “Using Hidden markov models to evaluate the risks of intrusions: System architecture and model validation,” Lect. Notes Comput. Sci., pp. 145–164, 2006.

[82] D. Ariu, R. Tronci, and G. Giacinto, “HMMPayl: An intrusion detection system based on hidden Markov models,” Comput. Secur., vol. 30, no. 4, pp. 221–241, 2011.

[83] S. S. Joshi and V. V. Phoha, “Investigating hidden Markov models capabilities in anomaly detection,” in Proc. ACM 43rd Annu. Southeast Reg. Conf., 2005, vol. 1, pp. 98–103.

[84] P. Dempster, N. M. Laird, and D. B. Robin, “Maximum likelihood from incomplete data via the EM algorithm,” J. Roy. Statist. Soc., Series B (methodological), pp. 1–38, 1977.

[85] W. W. Cohen, “Fast effective rule induction,” in Proc. 12th Int. Conf. Mach. Learn., Lake Tahoe, CA, USA, 1995, pp. 115–123.

[86] R. Michalski, “A theory and methodology of inductive learning,” Mach. Learn., vol. 1, pp. 83–134, 1983.

[87] W. Lee, S. Stolfo, and K. Mok, “A data mining framework for building intrusion detection models,” in Proc. IEEE Symp. Secur. Privacy, 1999, pp. 120–132.

[88] W. Fan, M. Miller, S. Stolfo, W. Lee, and P. Chan, “Using artificial anomalies to detect unknown and known network intrusions,” Knowl. Inf. Syst., vol. 6, no. 5, pp. 507–527, 2004.

[89] I. H. Witten and E. Frank, Data Mining: Practical Machine Learning Tools and Techniques, 3rd ed. San Mateo, CA, USA: Morgan Kaufmann, 2011.

[90] M. Panda and M. R. Patra, “Network intrusion detection using Naive Bayes,” Int. J. Comput. Sci. Netw. Secur., vol. 7, no. 12, pp. 258–263, 2007.

[91] N. B. Amor, S. Benferhat, and Z. Elouedi, “Naïve Bayes vs. decision trees in intrusion detection systems,” in Proc ACM Symp. Appl. Comput., 2004, pp. 420–424.

[92] R. Agrawal and R. Srikant, “Mining sequential patterns,” in Proc. IEEE 11th Int. Conf. Data Eng., 1995, pp. 3–14.

[93] Y. Hu and B. Panda, “A data mining approach for database intrusion detection,” in Proc. ACM Symp. Appl. Comput., 2004, pp. 711–716.

[94] Z. Li, A. Zhang, J. Lei, and L. Wang, “Real-time correlation of network security alerts,” in Proc. IEEE Int. Conf. e-Business Eng., 2007, pp. 73–80.

[95] V. Vapnik, The Nature of Statistical Learning Theory. New York, NY, USA: Springer, 2010.

[96] Y. Li, J. Xia, S. Zhang, J. Yan, X. Ai, and K. Dai, “An efficient intrusion detection system based on support vector machines and gradually feature removal method,” Expert Syst. Appl., vol. 39, no. 1, pp. 424–430, 2012.

[97] F. Amiri, M. Mahdi, R. Yousefi, C. Lucas, A. Shakery, and N. Yazdani, “Mutual information-based feature selection for IDSs,” J. Netw. Comput. Appl., vol. 34, no. 4, pp. 1184–1199, 2011.

[98] W. J. Hu, Y. H. Liao, and V. R. Vemuri, “Robust support vector machines for anomaly detection in computer security,” in Proc. 20th Int. Conf. Mach. Learn., 2003, pp. 282–289.

[99] C. Wagner, F. Jérôme, and E. Thomas, “Machine learning approach for IP-flow record anomaly detection,” in Networking 2011. New York, NY, USA: Springer, 2011, pp. 28–39.

[100] D. Brauckhoff, A. Wagner, and M. May, “Flame: A low-level anomaly modeling engine,” in Proc. Conf. Cyber Secur. Exp. Test, 2008.

[101] T. Shon and J. Moon, “A hybrid machine learning approach to network anomaly detection,” Inf. Sci., vol. 177, no. 18, pp. 3799–3821, Sep. 2007.

[102] T. Kohonen, Self-Organizing Map. New York, NY, USA: Springer, 1995.

[103] V. Paxson. (2004). Bro 0.9 [Online]. Available: [http://bro-ids.org](http://bro-ids.org/), accessed on Jun. 2014.

[104] R. Caruana and A. Niculescu-Mizil, “An empirical comparison of supervised learning algorithms,” in Proc. ACM 23rd Int. Conf. Mach. Learn., 2006, pp. 161–168.

[105] J. Dean and S. Ghemawat, “MapReduce: Simplified data processing on large clusters,” Commun. ACM, vol. 51, no. 1, pp. 107–113, 2008.

[106] H. Guang-Bin, D. H. Wang, and Y. Lan, “Extreme learning machines: A survey,” Int. J. Mach. Learn. Cybern., vol. 2, no. 2, pp. 107–122, 2011.

[107] K. Jain, J. Mao, and K. M. Mohiuddin, “Artificial neural networks: A tutorial,” Computer, vol. 29, no. 3, pp. 31–44, 1996.

[108] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo, “Fast discovery of association rules,” Adv. Knowl. Discov. Data Min., vol. 12, no. 1, pp. 307–328, 1996.

[109] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise,” Knowl. Discov. Data Min., vol. 96, pp. 226–231, 1996.

[110] P. S. Oliveto, J. He, and X. Yao, “Time complexity of evolutionary algorithms for combinatorial optimization: A decade of results,” Int. J. Autom. Comput., vol. 4, no. 3, pp. 281–293, 2007.

[111] G. D. Forney, “The Viterbi algorithm,” Proc. IEEE, vol. 61, no. 3, pp. 268–278, Mar. 1973.

[112] J. C. Burges, “A tutorial on support vector machines for pattern recognition,” Data Min. Knowl. Discov., vol. 2, no. 2, pp. 121–167, 1998.

[113] K. Ahsan and D. Kundur, “Practical data hiding in TCP/IP,” in Proc. ACM Multimedia Secur. Workshop, 2002, vol. 2, no. 7.

Anna L. Buczak photograph and biography not available at the time of Publication.

Erhan Guven photograph and biography not available at the time of Publication.