УДК 004.056.5:378.3

И.А. Ветров, В.В. Подтопельный

Особенности построения нейросетей с учетом специфики их обучения для решения задач поиска сетевых атак

Рассматриваются проблемы построения нейросетей для решения задач обнаружения сетевых вторжений с учетом современных общедоступных технологий. Анализируется несколько конфигураций нейросетей: простой персептрон, комбинированная сеть, состоящая из двух взаимосвязанных сетей, упрощенные сети на основе простого персептрона, LSTM-сети, использующие скрытые слои с функцией сжатия данных. Рассматриваются слабые и сильные стороны нейросетевых архитектур с учетом специфики их обучения на основе датасетов аномального трафика в задачах обнаружения вторжений.

Ключевые слова: сетевая атака, нейросеть, датасет, матрица признаков, функция активации, язык программирования Python.

DOI: 10.21293/1818-0442-2023-26-2-42-50

Современные разрабатываемые механизмы информационной безопасности с элементами искусственного интеллекта использует методы машинного обучения, позволяющие в автоматизированном режиме обучить систему распознавания угроз (в данном случае сетевых) без необходимости в затратах на настройку со стороны человека. Машинное обучение также позволит в автоматизированном режиме определить признаки, присущие нормальному и аномальному состоянию ЛВС.

Процесс создания и обучения систем анализа сетевых данных на основе машинного обучения (нейронных сетей) предполагает следующую последовательность этапов:

1. Предварительная обработка данных. На этом этапе отбрасываются повторяющиеся значения в наборах данных. Преобразовываются строковые значения в числовые для возможности обработки их в удобном формате для обучения нейронной сети. Удаляются строки набора данных, содержащие нулевые значения.

2. Оценка значимости и отбор признаков. Из набора данных отбрасываются признаки, несущие значения, которые может подделать злоумышленник, а также значения, имеющие сильные связи между собой и вследствие этого воспринимаемые нейронной сетью как дубликаты. Затем исключаются признаки, которые имели сильные связи по результатам корреляционного анализа.

3. Подготовка набора данных к обучению. На этом этапе атаки объединяются в массив аномалий (это необходимо для того, чтобы нейронная сеть могла обучиться определять нормальный трафик от аномального). Также на этом этапе масштабируются данные для корректного обучения нейронной сети.

4. Построение модели нейронной сети. На этом этапе необходимо определять размерность входных и выходных слоев, экспериментально подбирать размерность скрытых слоев путем множества попыток обучения либо путем эмпирического исследования пригодности численности слоев и нейронов в слоях за счет оценки достоверности, наименьшей ошибочности результатов. Также на этом этапе определяются параметры активации нейронной сети и параметры завершения обучения. В качестве корректировщика значений в процессе обучения выбирается оптимизатор (обычно используется «Adam») [1].

При создании нейронных сетей в настоящее время часто используют язык программирования Python, а также подготовленные с применением этого языка средства работы с массивами данных. Принимая во внимание распространённость подобных средств и инструментов, в настоящем исследовании было решено использовать следующее:

1. NumPy – библиотека с открытым исходным кодом для быстрой обработки многомерных массивов. Она написана на языке программирования C, что и обеспечивает быстродействие расчетов больших объемов данных.

2. Pandas – является пакетом для интерактивного анализа данных. Он особенно полезен для манипулирования данными реального мира с помощью комбинирования матричной математики NumPy и возможности обработки таблиц и реляционных баз данных.

3. Matplotlib – библиотека необходима для создания двумерных диаграмм и графиков, а также имеет возможность визуализировать данные для лучшего восприятия обрабатываемой информации.

4. TensorFlow – это высокоуровневый API для построения и обучения моделей глубокого обучения.

5. Seaborn – это библиотека визуализации данных Python, основанная на matplotlib. Он предоставляет собой высокоуровневый интерфейс для рисования привлекательных и информативных статистических графиков.

При всем многообразии нейросетевых технологий разработчики зачастую используют одни и те же инструменты и подходы к построению и обучению нейросетей. Однако данный подход приводит к технологической ограниченности. Требуется выявить данные ограничения и определить специфику их влияния на результат (точность) работы нейросети при анализе трафика на предмет выявления сетевых атак.

Также требуется определить наилучшие способы работы с нейросетями в заданных условиях. Для решения указанных задач рассмотрим особенности следующих нейросетевых архитектур, применяемых для обнаружения сетевых атак:

1. Архитектура на основе персептрона.

2. Последовательное использование двух нейросетей (двухкомпонентная сборка) для решения одной задачи со множеством входных данных.

3. Сети со сложной архитектурой, возможностью сохранения данных при обучении (LSTM).

Исходные данные для нейросетей

Для обучения нейросети поиска сетевых атак был выбран набор данных CICIDS2017 – это симуляция данных сетевого сообщения, разработанная в 2017 г. Канадским институтом компьютерной безопасности. В состав набора входят сообщение между 25 пользователями рабочей среды, двумя серверами, а также атаки, производимые на эту среду. Типы и количества атак представлены в табл. 1 (BENIGN – запись, не содержащая атаку) [2].

Таблица 1

Количественный состав набора данных CICIDS2017

	Тип записи	Количество записей
1.	BENIGN	2359087
2.	DoS Hulk	231072
3.	PortScan	158930
4.	DDoS	41835
5.	DoS GoldenEye	10293
6.	FTP-Patator	7938
7.	SSH-Patator	5897
8.	DoS slowloris	5796
9.	DoS Slowhttptest	5499
10.	Bot	1966
11.	Infiltration	36
12.	Heartbleed	11
13.	Web Attack – Brute Force	1507
14.	Web Attack – XSS	652
15.	Web Attack – SQL Injection	21

Предварительно сразу стоит отметить некоторые проблемы, возникающие при работе с датасетом. Первая проблема связана с составлением сигнатурных датасетов. Полных отечественных датасетов для сетевых атак сейчас в открытом доступе нет. Вторая проблема состоит в том, что нет в наличии отечественного генератора признаков. Для выделения признаков из захваченного пакета применяется генератор признаков CICFlowMetter (ранее известный как ISCXFlowMeter), созданный на языке Python, – генератор и анализатор двухпотокового трафика Ethernet для обнаружения аномалий. Это единственный, находящийся в открытом доступе генератор признаков.

Следует учитывать, что в заимствованном датасете неравномерно распределены количественные показатели сетевых атак. Это вызовет дисбаланс при обучении нейросети.

Предварительная обработка данных предполагает следующие этапы:

1) установка соответствия признаков и числовых значений;

2) изъятие всех маркеров с пустым значением (null) в наборе;

3) изъятие маркера «Fwd Header Length_copy» (признаки «Fwd Header Length» и «Fwd Header Length_copy» являются идентичными).

Различия в оценке значимости и отбор признаков

Для двухкомпонентной нейростевой сборки на основе простого персептрона предполагается следующее: будет произведен корректный выбор наиболее значимых параметров (обычно производится сокращение числа признаков), т.е. будет сформировано признаковое пространство, достаточное для решения поставленной задачи обнаружения сетевых атак [3]. В данном случае оно содержит следующие общие для многих атак маркеры (признаки): «Flow ID», «Source IP», «Source Port», «Destination IP», «Destination Port», «Protocol», «Timestamp» [3].

Рассмотрим сокращение признакового пространства. Подбор признаков для модели обнаружения DoS-атак предусматривает сокращение признакового пространства. Для двухкомпонентной нейросетевой сборки на основе простого персептрона после исключения признаков с наименьшей значимостью признаковое пространство сокращено до объединения 10 признаков [3]:

1. Средняя длина поля данных пакета TCP/IP (далее – длина пакета «Average Packet Size»).

2. Минимальное значение межпакетного интервала (IAT, inter-arrival time) в прямом направлении («Fwd IAT Min»).

3. Суммарная длина заголовков пакетов, переданных в прямом направлении («Fwd Header Length»).

4. Скорость потока данных («Flow Bytes/s»).

5. Максимальная длина переданного в прямом направлении пакета («Fwd Packet Length Max»).

6. Средняя длина переданных в прямом направлении пакетов («Fwd Packet Length Mean»).

7. Среднеквадратическое отклонение значения межпакетного интервала в прямом направлении пакетов («Fwd IAT Std»).

8. Среднее значение межпакетного интервала («Flow IAT Mean»).

9. Максимальная длина пакета («Max Packet Length»).

10. Суммарная длина переданных в прямом направлении пакетов («Total Length of Fwd Packets»).

Нужно отметить, что рассматриваемая нейросеть ориентирована на выявление не более трех атак при их имитации в клиентско-серверных структурах [4]. Для модели нейросети с ограниченным набором задач признаки, не вошедшие в 20 наиболее важных для обучения при работе встроенного механизма, были исключены, вследствие чего остались [5]:

1. Fwd Pkt Len Mean (средний размер пакета в прямом направлении) – 0,074.

2. Fwd Seg Size Avg (средний размер сегмента в прямом направлении) – 0,066.

3. Fwd Pkt Len Std (размер стандартного отклонения пакета в прямом направлении) – 0,046.

4. Flow Duration (всего пакетов в обратном направлении) – 0,040.

5. Flow IAT Max (максимальное время между двумя потоками) – 0,038.

6. Fwd Pkts/s (количество пакетов в секунду) – 0,038.

7. Bwd Pkts/s (количество обратных пакетов в секунду) – 0,037.

8. Flow IAT Mean (среднее время между двумя потоками) – 0,035.

9. Fwd IAT Min (минимальное время между двумя пакетами, отправленными в прямом направлении) – 0,034.

10. Subflow Fwd Byts (среднее количество байтов в подпотоке в прямом направлении) – 0,034.

11. Fwd IAT Mean (среднее время между двумя пакетами, отправленными в прямом направлении) – 0,033.

12. Flow Pkts/s (скорость потока пакетов, т.е. количество пакетов, переданных в секунду) – 0,031.

13. Fwd IAT Max (максимальное время между двумя пакетами, отправленными в прямом направлении) – 0,030.

14. Pkt Size Avg (средний размер пакета) – 0,030.

15. TotLen Fwd Pkts (общий размер пакетов в прямом направлении) – 0,030.

16. Flow IAT Min (минимальное время между двумя потоками) – 0,030.

17. Fwd IAT Tot (общее время между двумя пакетами, отправленными в прямом направлении) – 0,029.

18. Pkt Len Mean (средняя длина пакета) – 0,026.

19. Subflow Bwd Pkts (среднее количество пакетов в подпотоке в обратном направлении) – 0,020.

20. Flow Byts/s (скорость потока в байтах, т.е. количество пакетов, передаваемых в секунду) – 0,020.

Рядом с каждым признаком расположена числовая оценка значимости.

В процессе детального изучения оставшихся признаков было замечено, что многие из них имеют зависимость друг от друга. Для исключения корреляции было решено составить корреляционную матрицу двадцати наиболее важных для обучения признаков.

Матрица была составлена с использованием библиотек seaborn и matplotlib.

На рис. 1 представлена корреляционная матрица с коэффициентами корреляции Пирсона для этих признаков.

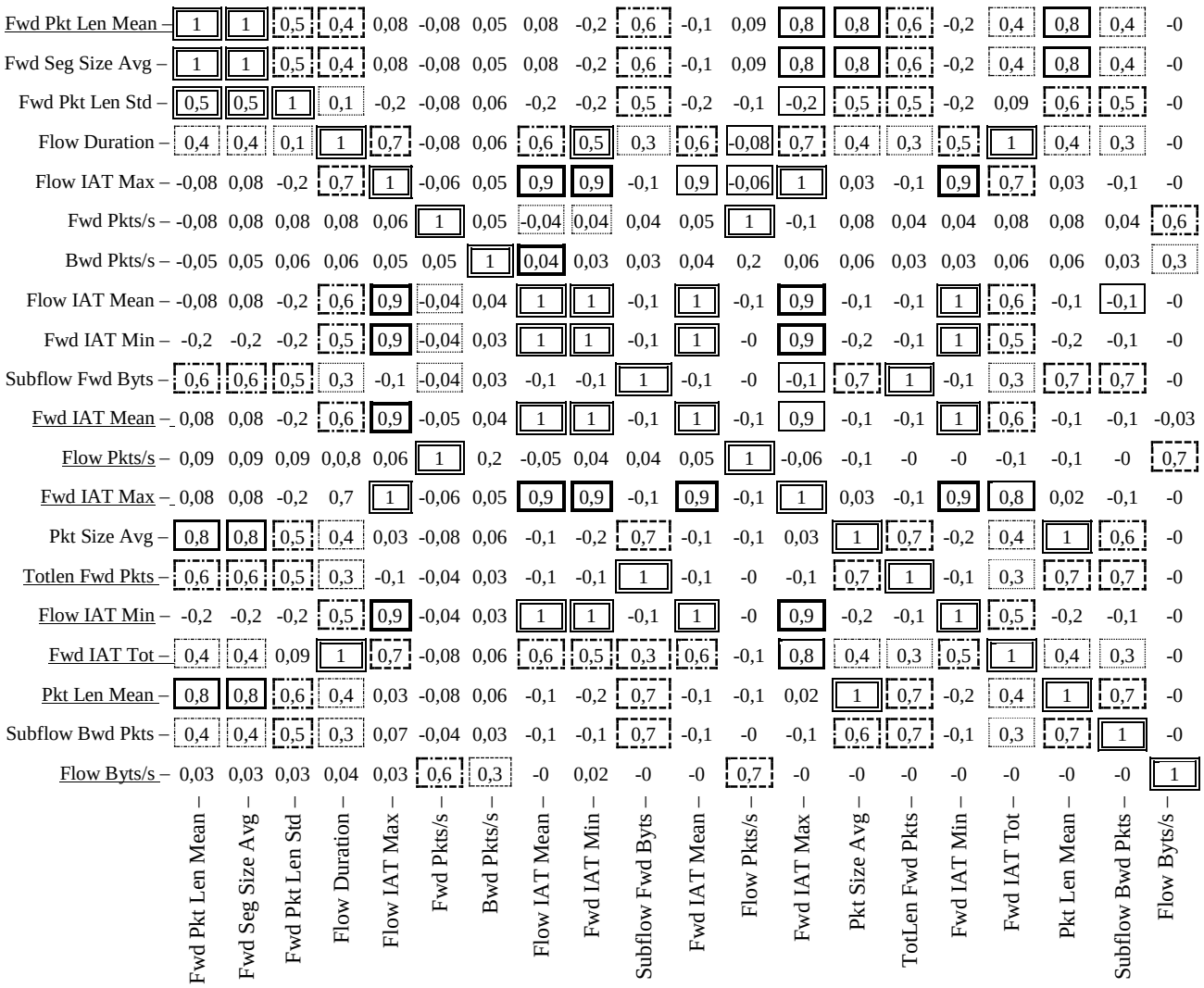

Рис. 1. Матрица корреляции признаков

В приведенной матрице степень корреляции признаков соответствует типу линий ячейки со значением коэффициента: выделение пунктиром означает слабую корреляцию, выделение линиями указывает на прирост степени корреляции, сильная корреляция показана в ячейках, обведенных жирными линиями, в обведенных двойными линиями – максимально сильные корреляции.

В общей матрице корреляции на рис. 1 подчеркиванием отмечены признаки, коэффициент корреляции которых с каким-либо другим признаком оказался равен единице.

Были исключены следующие признаки:

1. Flow Pkts/s.
2. TotLen Fwd Pkts.
3. Fwd IAT Min.
4. Fwd IAT Tot.
5. Flow IAT Min.
6. Fwd IAT Mean.
7. Pkt Len Mean.
8. Fwd Seg Size Avg.
9. Fwd IAT Max.
10. Flow Byts/s.

На рис. 2 приведена корреляционная матрица, составленная для оставшихся признаков.

	Fwd Pkt Len Mean –	Fwd Pkt Len Std –	Flow Duration –	Flow IAT Max –	Fwd Pkts/s –	Bwd Pkts/s –	Flow IAT Mean –	Subflow Fwd Byts –	Pkt Size Avg –	Subflow Bwd Pkts –
Fwd Pkt Len Mean –	1	0,5	0,4	0,08	-0,08	-0,05	-0,08	0,6	0,8	0,4
Fwd Pkt Len Std –	0,5	1	0,1	-0,2	-0,08	-0,06	-0,2	0,5	0,5	0,5
Flow Duration –	0,4	0,1	1	0,7	-0,08	-0,06	0,6	0,3	0,4	0,3
Flow IAT Max –	0,08	-0,2	0,7	1	-0,06	-0,05	0,9	-0,1	0,03	-0,07
Fwd Pkts/s –	-0,08	-0,08	-0,08	-0,06	1	0,05	-0,04	-0,04	-0,08	-0,04
Bwd Pkts/s –	-0,05	-0,06	-0,06	-0,05	0,05	1	-0,04	-0,03	-0,06	-0,03
Flow IAT Mean –	-0,08	-0,2	0,6	0,9	-0,04	-0,04	1	-0,1	-0,1	-0,1
Subflow Fwd Byts –	0,6	0,5	0,3	-0,1	-0,04	-0,03	-0,1	1	0,7	0,7
Pkt Size Avg –	0,8	0,5	0,4	0,03	-0,08	-0,06	-0,1	0,7	1	0,6
Subflow Bwd Pkts –	0,4	0,5	0,3	-0,07	-0,04	-0,03	-0,1	0,7	0,6	1

Рис. 2. Матрица корреляции оставшихся признаков

Как видно, оставшиеся признаки не имеют прямой корреляции друг с другом.

Оставшиеся признаки:

1. Fwd Pkt Len Mean.
2. Fwd Pkt Len Std.
3. Flow Duration.
4. Flow IAT Max.
5. Fwd Pkts/s.
6. Bwd Pkts/s.
7. Flow IAT Mean.
8. Subflow Fwd Byts.
9. Pkt Size Avg.
10. Subflow Bwd Pkts.

После обработки входного набора данных в самом наборе осталось 3094 записи, из которых 928 записей класса «есть атака» и 2166 записей класса «нет атаки». Признаковое пространство было сокращено до десяти наиболее важных для обучения, напрямую не коррелирующих друг с другом.

Рассмотрим подбор признаков для нейросетей, разработанных на основе модели LSTM. В данном случае выходные значения (после работы нейросети) представлены в формате файла CSV с шестью столбцами, помеченными для каждого потока, а именно FlowID, SourceIP, DestinationIP, SourcePort, DestinationPort и Protocol с более чем 80 функциями сетевого трафика.

Для ускорения обучения и по причине несбалансированности (количество записей по разным видам атак сильно разнится) имеющийся датасет прошел предобработку. Для этого использовались алгоритм оверсемплинга (оверсемплинг – это процесс генерации синтетических данных, который пытается случайным образом сгенерировать выборку атрибутов из наблюдений в классе меньшинства) в технике SMOOTE для атак типа Bot, Brute Force, Web и сетевая разведка. Для остальных типов атак, где количество записей довольно большое, данные масштабировались, т.е. часть данных просто была отброшена [6].

Также для того чтобы приблизиться к обнаружению атак в реальном времени, необходимо ограничить набор признаков, использующихся для классификации типов атак. Признаки, использующиеся в предобработанном датасете, были обозначены как базовые:

– tot_fw_pk – всего пакетов в прямом направлении;

– tot_bw_pk – всего пакетов в обратном направлении;

– fw_pkt_s – количество пересылаемых пакетов в секунду;

– protocol;

– port;

– IP-адрес.

Результаты обнаружения атак при исследовании на полном наборе и на базовых признаках представлены в табл. 2.

Нейронные сети обращают больше внимания на аргументы, имеющие большие значения в сравнении с другими данными. Для того чтобы сеть обучалась корректно, необходимо масштабировать данные. В наборе оставшиеся только числовые данные можно преобразовать с помощью математических функций, в итоге получив значения в промежутке (–1, 1).

Таблица 2

Точность обнаружения атак при разных наборах признаков

№ п/п	Трафик	Точность обнаружения (все признаки), %	Точность обнаружения (базовые признаки), %
1	Нормальный трафик	96,71	96,23
2	Brute force	93,76	94,06
3	Web	82,14	83,86
4	DDoS	97,9	99,21
5	DoS	78,96	89,66
6	Bot	99,86	99,21
7	Сетевая разведка	93,58	95,31

Определение специфики процесса обучения и использования метрик при обучении нейросетей обнаружения сетевых вторжений

В процессе обучения нейронной сети можно столкнуться с проблемой переобучения. Переобучение – это излишне точное соответствие нейронной сети конкретному набору обучающих примеров, при котором сеть теряет способность к обобщению.

При обучении модели были использованы следующие подходы для того, чтобы избежать проблемы переобучения:

1. Разделение обучающей выборки на три части: обучающая выборка, выборка валидации и тестовая выборка.

2. Остановка обучения при малом изменении весовых коэффициентов.

3. Выбор функции активации, позволяющей эффективно обучать нейронную сеть.

4. Использование наименьшего возможного числа нейронов в слоях.

Рассмотрим специфику обучения для двухкомпонентной нейросетевой сборки на основе простого персептрона. Модель нейронной сети представляет собой многослойный персептрон – сеть прямого распространения, где все связи направлены строго от входных нейронов к выходным. В составе приводимой нейронной сети присутствуют:

– входной слой (он содержит десять нейронов);

– три скрытых слоя (первый скрытый слой содержит пятьдесят нейронов, второй содержит десять нейронов, третий содержит один нейрон);

– выходной слой с одним нейроном.

Для последнего слоя в качестве функции активации используется активатор Softmax (при программировании реализуется методом «Winner Takes All») [6].

Промежуточные слои используют операцию ReLu в качестве функции активации (также называемой фактором нелинейности). Функция является кусочно-линейной и используется для преобразования отрицательных значений в ноль. Она необходима для того, чтобы предотвратить затухание градиента в процессе обучения.

В качестве оценки для корректировки значений весов с целью уменьшения потерь использовался оптимизатор Adam (adaptive moment estimation – адаптивная оценка момента). Алгоритм Adam использует импульс в виде оценки первого момента (с экспоненциальными весами) градиента, также он осуществляет поправку на смещение в оценке как первых моментов (член импульса), так и вторых (нецентрированных) моментов для учета их инициализации в начале координат [6].

При обучении нейронной сети важной задачей было избегать ее переобучения. Импортируемая библиотека Keras включает в себя функцию ранней остановки для достижения поставленной цели. Метод ранней остановки позволяет задать неограниченное количество эпох обучения и указать пороговое значение производительности, при превышении которого нейросеть прекратит обучение.

Существует другой подход к обучению, он предполагает, что функция ранней остановки контролирует количество эпох обучения. Недостающей составляющей остается поиск оптимального количество скрытых слоев и количество нейронов в них.

Наиболее оптимальным количеством слоев считается такой результат обучения, в котором значение потерь стремится к нулю, а значение точности стремится к единице.

Крайним пороговым значением количества скрытых слоем считается такое значение, при котором однозначно можно определить, что тенденция дальнейшего увеличения количества скрытых слоев не приведет к получению лучших результатов. Согласно этому утверждению было принято решение остановиться на этапе с порядковым номером № 27. Результирующие данные поиска оптимального количества скрытых слоев представлены в табл. 3.

Начиная с использования 15 слоев и более наблюдается тенденция резкого снижения точности ИНС и больших потерь относительно предыдущих вариантов.

Наилучшая оценка обучения ИНС получена при использовании шести скрытых слоев, поскольку были получены наилучшие показатели в плане минимизации потерь при повышении точности.

После определения количества скрытых слоев был проведен анализ оптимального количества

нейронов в скрытых слоях. Тестирование проводилось в диапазоне от 1 до 50 нейронов.

В результате анализа оптимального количества нейронов в скрытых слоях было установлено, что лучший результат был достигнут в попытке с использованием не более 40 нейронов в каждом слое. Таким образом, модель будет иметь 6 скрытых слоев и в каждом скрытом слое будет находиться по 24 (при данном количестве нейронов наименьшее количество потерь) нейрона. Точность модели достигла более 98%.

Рассмотрим особенности модели нейросети с ограниченным набором задач (выявление не более трех атак). Нейросеть для ограниченного количества специфичных (web-атак) атак (Brute Force, XSS, SQL Injection). Было принято решение работать либо с CICIDS2017, либо CSE-CIC-IDS2018 в связи с их актуальностью [7].

Таблица 3

Подбор оптимального количества скрытых слоев

№ п/п	Количество нейронов в скрытых слоёв	Потери	Точность
1	2	3	4
1	1	0,68335	0,56994
2	2	0,68346	0,56994
3	3	0,02298	0,99772
4	4	0,00729	0,99863
5	5	0,01075	0,99849
6	6	0,00679	0,99883
7	7	0,00722	0,99854
8	8	0,00806	0,99865
9	9	0,00736	0,99867
10	10	0,01158	0,99858
11	11	0,01019	0,99860
12	12	0,00678	0,99865
13	13	0,01083	0,99856
14	14	0,00699	0,99865
15	15	0,00533	0,99865
16	16	0,01168	0,99847
17	17	0,00521	0,99876
18	18	0,00540	0,99883
19	19	0,00709	0,99874
20	20	0,00946	0,99825
21	21	0,00693	0,99883
22	22	0,00572	0,99874
23	23	0,00922	0,99807
24	24	0,00543	0,99894
25	25	0,00979	0,99852
26	26	0,00684	0,99874
27	27	0,00492	0,99883

После детального рассмотрения было обнаружено, что в наборе CICIDS2017 содержатся пустые записи, повтор признаков, специальные символы, усложняющие обработку. Также стоит отметить, что данный набор был составлен из записей трафика одной сети, что негативно скажется на работе разработанной системы в других сетях, так как многие признаки, содержащиеся в наборе, зависят от физической структуры сети.

Набор данных содержит информацию, полученную при помощи программного обеспечения – анализатора сетевого трафика CICFlowMeter V3 (и представлен в виде файла с расширением csv).

Признаки исходного набора и их количество:

1. Нормальный трафик – 2097154.
2. Атаки типа Brute Force – 611.
3. Атаки типа XSS – 230.
4. Атаки типа SQL Injection – 87.
5. Всего записей – 2098082.

Подготовленная выборка является несбалансированной: при количестве записей 2098082 класс «нет атаки» объединяет 2097154 записей, в то время как класс «есть атака» – всего 928 записей. Для избавления от дисбаланса классов был выбран метод случайного сэмплирования, также известный как субдискретизация [8]. Данный метод заключается в удалении случайно выбранных записей с пометкой класса «отсутствие атаки». Целевое соотношение количества записей с пометкой «нет атаки» и «есть атака» было выбрано 70 и 30% общего числа записей соответственно. Сэмплирование реализуется следующим образом. Задаются известные начальные параметры набора:

- attack_total = 928;
- benign_total = 2097154;
- enlargement = 1,1.

После проведения сэмплирования дисбаланса классов было исключено 2094988 записей. Количественные показатели оставшихся признаков обработанного набора:

1. Нормальный трафик – 2166.
2. Атаки типа Brute Force – 611.
3. Атаки типа XSS – 230.
4. Атаки типа SQL Injection – 87.
5. Всего записей – 3094.

Набор данных содержит признаковое пространство из 80 признаков, из которых предварительно были исключены 'Dst Port' (порт назначения), 'Protocol' (протокол), 'Timestamp' (временная метка) т.к. они относительно легко могут быть подделаны злоумышленником, а значит, не должны принимать участие в процессе обучения.

Далее, был проведен анализ значимости признаков с помощью встроенного механизма метода sklearn.ensemble.RandomForestClassifier (атрибут feature_importances_).

Рассмотрим особенности для двухкомпонентной нейростевой сборки на основе простого персептрона. Чтобы определить количество эпох, была использована функция EarlyStopping библиотеки Keras (решение оптимального набора слоев и количества нейронов). Эта функция останавливает обучение, когда отслеживаемый параметр перестает улучшаться [9]. В рассматриваем случае функция обладает следующими аргументами:

a) «monitor» – в нейронной сети отслеживаемое значение «val_loss», этот параметр вычисляет количество ошибочных вычислений на тестовых данных (требуется, чтобы этот параметр стремился к 0);

b) «min_delta» – разница между значениями, которая приведет к остановке обучения, – равен 0,001;

c) «patience» – количество эпох, отведенное для ожидания улучшения результата в случае фиксировании разницы результата на 0,001 за 5 эпох, прекращает обучение;

d) «mode» – этот параметр обладает 3 стандартными значениями:

– «min» – режим отвечает за контроль уменьшения значения;

– «max» – режим отвечает за контроль увеличения значения;

– «auto» – режим, который выбирает направление в зависимости от имени контролируемого значения.

Так как отслеживается параметр ошибок, контролируемое значение должно уменьшаться.

Для обучения модели предварительно было задано 100 эпох, но обучение было прервано на 30-й эпохе. При этом показатель ошибочных ответов оказался равным 0,0318. Соответственно, точность на тестовом наборе составила 96,82%.

Для оценки эффективности того, как обученная модель различает аномальные и нормальные данные, использовались метрики ROC-AUC. Кривая ROC определяет долю истинно положительных классификаций (TRP) по отношению к доле ложноположительных классификаций (FPR (доля FPR – это пропорция отрицательных образцов, которые были некорректно классифицированы как положительные)) [10]. Она равна единице минус доля истинно отрицательных классификаций (TNR), представляющая собой пропорцию отрицательных образцов, которые были корректно классифицированы как отрицательные.

На рис. 3 продемонстрированы результаты расчета оценочных значений.

```
Accuracy score: 0.9888651126176877
====================================
Recall score: 0.9888651126176877
====================================
Precision score: 0.9888401477035845
====================================
F1 score: 0.9887749664800025
====================================
ROC-AUC score: 9720305423054391
```

Рис. 3. Оценки эффективности

Определение специфики влияния принципов формирования нейросетей обнаружения сетевых вторжений на точность результатов при их обучении

В целом отбор и корреляция признаков из-за своеобразия датасета и способов отбора признаков (ручного отбора) сводится к сокращению пространства признаков. Метод семплинга при работе со сложными архитектурами нейросетей является эффективным приемом для логичного смыслового увязывания статистических свойств выборки и цели моделирования. При этом семплинг позволяет увеличить размерность критериального пространства и одновременно выступает средством разрешения проблемы.

При использовании LSTM необходимо соблюдать баланс равенств входных данных по атакам и по нормальному трафику, поскольку дисбаланс в итоге приводит к потере эффективности работы нейросети из-за неправильного применения классификатора.

При этом простые сети (сети прямого распространения на основе простого персептрона) без постоянного обучения более стабильны, но требуют превентивного нового обучения при появлении новых атак (атак нулевого дня).

Для сетей прямого распространения следует отметить приоритетность деления датасета при обработке в слоях нейросети на данные по атакам с большим количеством записей и наименьшим количеством. При малом количестве записей обучение не эффективно, поскольку нейросеть из-за дисбаланса в количестве записей сводит атаки с наименьшим количеством к статистической погрешности. Здесь целесообразно разделение датасетов по принадлежности к определенным типам атак. Таким образом, создается несколько нейросетей для одной-двух схожих атак. Другой способ нивелирования дисбаланса количества записей предполагает насыщение датасета дополнительными записями, относящимися к атакам с наименьшим количеством записей.

Для сложных сетей типа LSTM наилучшим вариантом является разделение датасета на две выборки: нормальный трафик, аномальный трафик (при этом требуется соблюдать процентное соотношение записей по скомпрометированному трафику и нормальному). Однако для реализации постоянного процесса обнаружения сетевых угроз подобная сеть непригодна, поскольку при постоянном обучении сеть обучится неправильно. Персептрон в этом отношении для решения задач стабилен – его ответы со временем не будут меняться.

С учетом заданного инструментария (и, следовательно, их технологических ограничений), а также устоявшегося алгоритма работы построения нейросетей для решения задач в области отслеживания трафика следует отметить следующее: существует несколько способов достижения точности полученных результатов. Каждый из способов характеризуется своеобразием обработки входных данных и интерпретации полученных результатов, обучения нейросети (в плане рассматриваемой точности обучения).

Первый способ предполагает использование поиска наиболее точного результата и обучения с помощью перебора количественного состава слоёв и нейронов в слоях. При этом фиксируется наиболее оптимальное соотношение точности и архитектурных параметров нейросети. Такой подход характерен при построении нейросети на основе модели простого персептрона.

Однако усложнение модели подобного типа за счёт внедрения слоёв с более специфическими функциями (наподобие слоев drop-out) не приводит к повышению точности при обучении. Таким образом, вычисление оптимальных характеристик модели нейросети приводит к ограничению ее архитектурных особенностей. Более того, усложнение алгоритма анализа приводит к увеличению количества времени, которое требуется на перебор массива

входящих показателей при обучении. При практическом использовании нейросетей для обнаружения атак подобное явление будет вызывать запаздывание фиксации вредоносного трафика.

Второй способ (использовался при работе с нейросетью с ограниченным набором задач) предполагает повышение точности при обучении с помощью модификации корреляционной матрицы признаков коэффициентами корреляции. Для повышения скорости обработки оптимальных значений матрицы предлагается способ деления входящих значений параметров сетевых атак при предварительной обработке данных на мелкие сегменты, включающие данные только нескольких выбранных из множества атакующих воздействий. Таким образом, предполагается создание некоторого массива нейросетей, каждая из которых обучена распознавать ограниченное число сетевых атак (предполагающих также сравнительно небольшое количество обучающих данных), что в итоге позволяет минимизировать ошибки.

Третий способ повышения точности, снимающий проблемы множества записей и несбалансированности для разных атак, предлагает наиболее упрощённый вариант работы с данными, а именно: предполагается последовательное использование множества записей выявления аномального и нормального состояния сетевого трафика. После получения первых результатов анализа трафика на предмет выявления аномалий производится перенаправление параметров аномального трафика для дальнейшей обработки в нейросети, которая определяет конкретный вид атаки. При этом отбор данных, признаков корреляционной матрицей осуществляется вручную (специалист самостоятельно должен вычленять «выбросы»). Необходимо подчеркнуть, что в этом случае вторая сеть, если она работает отдельно от первой, даёт меньшую точность (70%) в сравнении с сетью, работающей на основе распределения нормального и аномального трафика (95–97%) с учетом полной идентичности количества слоев и количества нейронов в слоях. При увеличении количества нейронов во второй сети, что нарушает их архитектурную идентичность, ее точность повышается. Из этого следует, что сочетание различных по архитектуре нейросетей дает большую точность в задачах классификации вредоносного трафика, нежели использование одной или сопряжённых двух нейросетей. Более того, минимизация ошибок во второй нейросети достигается за счёт предварительного отбора заранее полученных в результате работы первой сети скомпрометированных данных. Практически вторая сеть, ориентированная на классификацию типов атак, обучается на основе результатов работы первой нейросети (ориентирована на поиски аномального трафика).

Для нейросетей сложной архитектуры, предполагающей использование дроп-аут слоёв, построенной наподобие архитектур типа LSTM (с возможностью сохранения временных результатов работы), также требуется использование метода ручной обработки данных, а точнее, корреляции признаков, модификации признакового пространства.

И второй, и третий способ с точки зрения архитектуры построения и обучения нейросетей являются наиболее приемлемыми, поскольку позволяют реализовать требуемые архитектуры в соответствии с задачами, которые решает непосредственно специалист информационной безопасности.

Литература

1. Панченко А.А. Анализ подходов к построению системы защиты информации на базе модели процесса обработки данных / А.А. Панченко, М.В. Аникиенко, В.Н. Пржегорлинский // Вестник Рязанского гос. радиотехнического ун-та. – 2005. – № 16. – С. 120–123.

2. Горюнов М.Н. Синтез модели машинного обучения для обнаружения компьютерных атак на основе набора данных CICIDS2017 / М.Н. Горюнов, А.Г. Мацкевич, Д.А. Рыболовлев // Труды ИСП РАН. – 2020. – Т. 32, вып. 5. – С. 81–94.

3. Горюнов М.Н. Оценка применимости методов машинного обучения для обнаружения компьютерных атак / М.Н. Горюнов, А.А. Рыболовлев, Д.А. Рыболовлев // Информационные системы и технологии (Орел). – 2020. – № 6. – С. 103–111.

4. Гончаров В.А. Исследование возможностей противодействия сетевым информационным атакам со стороны защищенных ОС и систем обнаружения информационных атак / В.А. Гончаров, В.Н. Пржегорлинский // Вестник Рязанского гос. радиотехнического ун-та. – 2007. – № 20. – С. 10–14.

5. Гончаров В.А. Метод обнаружения сетевых атак, основанный на кластерном анализе взаимодействия узлов вычислительной сети / В.А. Гончаров, В.Н. Пржегорлинский // Вестник Рязанского гос. радиотехнического ун-та. – 2011. – № 36. – С. 3–10.

6. Жерон А. Прикладное машинное обучение с помощью Scikit-Learn и TensorFlow: концепции, инструменты и техники для создания интеллектуальных систем: пер. с англ. – Спб.: Альфа-книга, 2018. – 688 с.

7. Intrusion Detection Evaluation Dataset (CIC-IDS2018) [Электронный ресурс]. – Режим доступа: https://www.unb.ca/cic/datasets/ids-2018.html, свободный (дата обращения: 02.04.2022).

8. Leskovec J. Mining of Massive Datasets / J. Leskovec, A. Rajaraman, J. Ullman. – Cambridge: Cambridge University Press, 2014. – 511 p.

9. Domingos P.A. Few Useful Things to know about Machine Learning // Communications of the ACM. – 2012. – Vol. 55, No. 10. – P. 78–87.

10. Kostas K. Anomaly Detection in Networks Using Machine Learning. – Essex: School of Computer Science and Electronic Engineering University of Essex, 2018. – 70 p.

Ветров Игорь Анатольевич
Канд. техн. наук, доцент образовательно-научного кластера «Институт высоких технологий»
Балтийского федерального университета им. И. Канта
А. Невского ул., 14, г. Калининград, Россия, 236041
Тел.: +7-906-216-47-19
Эл. почта: vetrov.gosha2009@yandex.ru

Подтопельный Владислав Владимирович
Ст. преп. Института цифровых технологий (ИЦТ)
Калининградского государственного технического
университета (КГТУ)
Советский пр-т, 1, г. Калининград, 236022
ORCID: 0000-0002-7618-3224
Тел.: +7-900-353-98-81
Эл. почта: ionpvv@mail.ru

Vetrov I.A., Podtopelny V.V.

Features of building neural networks taking into account the specifics of their training to solve the tasks of searching for network attacks

The problems of building neural networks to solve the problems of detecting network intrusions, taking into account modern publicly available technologies, are considered. Several configurations of neural networks are analyzed: a simple perceptron, a combined network consisting of two interconnected networks, simplified networks based on a simple perceptron, LSTM networks using hidden layers with data compression function. The weaknesses and strengths of neural network architectures are considered, taking into account the specifics of their training based on abnormal traffic datasets in intrusion detection tasks.

Keywords: network attack, neural network, dataset, feature matrix, activation function.
DOI: 10.21293/1818-0442-2023-26-2-42-50

References

1. Panchenko A.A., Anikienko M.V., Przhegorlinsky V.N [Analysis of approaches to building an information security system based on a data processing model]. *Vestnik of Ryazan State Radioengineering University*, 2005, no. 16, pp. 120–123 (in Russ.).

2. Goryunov M.N., Matskevich A.G., Rybolovlev D.A. [Synthesis of a machine learning model for detecting computer attacks based on the CICIDS2017 dataset]. *Proceedings of the Institute for System Programming of the Russian Academy of Sciences*, 2020, vol. 32, iss. 5, pp. 81–93 (in Russ.).

3. Goryunov M.N., Rybolovlev A.A., Rybolovlev D.A. [Evaluation of the applicability of machine learning methods for detecting computer attacks]. *Information Systems and Technologies*, 2020, no. 6, pp. 103–111 (in Russ.).

4. Goncharov V.A., Przhegorlinsky V.N. [Investigation of the possibilities of countering network information attacks by protected operating systems and information attack detection systems]. *Vestnik of Ryazan State Radioengineering University*, 2007, no. 20, pp. 10–14 (in Russ.).

5. Goncharov V.A., Przhegorlinsky V.N. [Method of detecting network attacks based on cluster analysis of interaction of computer network nodes]. *Vestnik of Ryazan State Radioengineering University*, 2011, no. 36, pp. 3–10 (in Russ.).

6. Geron A. *Prikladnoe mashinnoe obuchenie s pomoshch'yu Scikit-Learn i TensorFlow: koncepcii, instrumenty i tekhniki dlya sozdaniya intellektual'nyh sistem* [Applied machine learning using the scikit package-learn and TensorFlow: concepts, tools and techniques for creating intelligent systems]. St. Petersburg, Alfa-book, 2018. 688 p. (in Russ.).

7. Intrusion Detection Evaluation Dataset (CIC-IDS2018). Available at: https://www.unb.ca/cic/datasets/ids-2018.html (Accessed: April 02, 2022).

8. Leskovets J., Rajaraman A., Ulman J. *Intelligent Analysis of Massive Data Sets*. Cambridge, Cambridge University Press, 2014, 476 p.

9. Domingos P. A few useful things you need to know about machine learning. *ACM Communications*, 2012, vol. 55, no. 10, pp. 78–87.

10. Kostas K. Anomaly detection in networks using machine learning. Essex, School of Computer Science and Electronic Engineering University of Essex, 2018, 70 p.

Igor A. Vetrov
Candidate of Sciences in Engineering, Associate Professor,
Institute of Physical and Mathematical Sciences and
Information Technologies, I. Kant Baltic Federal University
14, A. Nevsky st., Kaliningrad, Russia, 236041
Phone: +7-906-216-47-19
Email: vetrov.gosha2009@yandex.ru

Vladislav V. Podtopelny
Senior Lecturer, Institute of Digital Technologies, Federal
State Budgetary Educational Institution of Higher Education
Kaliningrad State Technical University
1, Sovetsky st., Kaliningrad, Russia, 236022
ORCID: 0000-0002-7618-3224
Phone: +7-900-353-98-81
Email: ionpvv@mail.ru