УДК 004.056; 004.93; 004.8

© 2020 г. **О.С. Амосов,** д-р техн. наук,
С.Г. Амосова, канд. техн. наук
(Институт проблем управления им. В.А. Трапезникова РАН, Москва),
Д.С. Магола, канд. техн. наук
(Комсомольский-на-Амуре государственный университет)

ИСПОЛЬЗОВАНИЕ ГЕНЕТИЧЕСКОГО АЛГОРИТМА ДЛЯ НАСТРОЙКИ ГЛУБОКИХ НЕЙРОННЫХ СЕТЕЙ В ЗАДАЧАХ КЛАССИФИКАЦИИ КОМПЬЮТЕРНЫХ АТАК

Дана постановка задачи многоклассовой сетевой классификации компьютерных атак. Для ее решения рассмотрена возможность использования технологии глубоких нейронных сетей. Выбрана архитектура глубокой нейронной сети на основе стратегии объединения набора сверточных и рекуррентных слоев LSTM. Предложена оптимизация параметров нейронной сети на основе генетического алгоритма. Представлены результаты моделирования, которые показывают возможность решения задачи сетевой классификации в реальном времени.

Ключевые слова: глубокая нейронная сеть, архитектура, параметры оптимизации, сверточный слой, рекуррентный слой, долгая краткосрочная память, сетевая классификация, компьютерная атака.

DOI: 10.22250/isu.2020.66.104-117

Введение

В настоящее время с помощью глубоких нейронных сетей (ГНС) получены положительные результаты для распознавания и классификации лиц, технических объектов, динамических ситуаций, информационной безопасности, прогнозирования временных рядов [1 – 4]. Однако нет строгой теории проектирования и структурно-параметрической оптимизации ГНС. При этом известны отдельные подходы для проектирования интеллектуальных систем, в том числе и для нейронных сетей [5 – 9].

Одной из стратегий построения ГНС является объединение скорости и легкости сверточных сетей с чувствительностью к порядку рекуррентных сетей – использование сверточной сети для предварительной обработки данных перед передачей их в рекуррентную сеть [10, 11].

Такая стратегия и взята для выбора архитектуры глубоких нейронных сетей с целью классификации компьютерных атак, а для оптимизации параметров использован генетический алгоритм. В настоящее время кроме традиционных методов оценивания стохастических процессов – таких как байесовский, небайесовский подходы и метод наименьших квадратов [1 – 3], предлагаются вычислительный метод и алгоритмы с использованием вейвлетов [4 – 9].

Постановка задачи сетевой классификации

Следуя [12], приведем постановку задачи многоклассовой классификации. Под объектом в дальнейшем будем понимать сетевую атаку или аномалию.

Дано: множество Ω, в котором хранится описание объектов $\omega = \Omega$, заданных признаками $x_i, i = \overline{1,n}$, совокупность которых для объекта ω представлена векторными описаниями $\mathbf{x} = \mathbf{\Phi}(\omega) = (x_1(\omega), x_2(\omega), ...x_i(\omega))^T$; множество классов $\mathbf{B} = \{\beta_1, \beta_2, ..., \beta_c\}$, c –количество классов; априорная информация, которая представлена обучающим множеством $\mathbf{D} = \{(\mathbf{x}^j, \beta^j)\}$, $j = \overline{1,L}$, заданным таблицей, каждая строка j которой содержит векторное описание объекта $\mathbf{x} = \mathbf{\Phi}(\omega)$ и метку класса $\beta_k, k = \overline{1,c}$. Заметим, что обучающее множество характеризует неизвестное отображение $^*\mathbf{F} : \Omega \to \mathbf{B}$.

Требуется по поступающим фрагментам $\mathbf{I}_t$ непрерывного сетевого трафика $\mathbf{V} = (\mathbf{I}_1, ..., \mathbf{I}_t, ..., \mathbf{I}_\tau)$ и априорной информации, заданной обучающим множеством $\mathbf{D} = \{(\mathbf{x}^j, \beta^j)\}$, $j = \overline{1,L}$ для глубокого обучения НС с учителем, решить задачу распознавания объектов: обнаружить объекты ω в виде оценки признаков $\tilde{\mathbf{x}}$ с помощью нейронных сетей, реализующих отображение $\mathbf{F}_1 : \mathbf{I}_t \to \tilde{\mathbf{x}}$, и классифицировать их с использованием отображения $\mathbf{F}_2 : \tilde{\mathbf{x}} \to \beta_k, k = \overline{1,c}$ в соответствии с заданным критерием $P(\tilde{\mathbf{x}})$, минимизирующим вероятность ошибки классификации.

Таким образом, необходимо найти отображение $\mathbf{F} : \mathbf{I}_t \to \beta_k, k = \overline{1,c}$, при котором $\mathbf{F}$ – является набором функций и нейросетевых алгоритмов $\mathbf{f}_i, i = \overline{1, N_f}$.

Решение задачи сетевой классификации

Решение задачи сетевой классификации предлагается осуществлять по следующему алгоритму на основе глубоких нейронных сетей.

1. Выделение из непрерывного сетевого трафика $\mathbf{V} = (\mathbf{I}_1,...,\mathbf{I}_t,...,\mathbf{I}_\tau)$ фрагмента $\mathbf{I}_t$ размером $w^{I_t} \times h^{I_t}$, где t – номер текущего фрагмента.

2. Преобразование фрагмента $\mathbf{I}_t$ размером $w^{I_t} \times h^{I_t}$ в фрагмент $\mathbf{I}'_t$ размером $k^{I'_t} \times s^{I'_t} \times h^{I_t}$, где $k^{I'_t} \times s^{I'_t}$ содержит все данные w^{I_t}. Другими словами, преобразование исходного двумерного тензора в трехмерный.

3. Извлечение информативных признаков $\mathbf{R}$ с использованием предобученной глубокой нейронной сети (NN) $\mathbf{\Phi}^{NN} : \mathbf{R} \to \tilde{\mathbf{x}}$, где $\tilde{\mathbf{x}}$ – данные фрагмента $\mathbf{I}_t$ переведенные в пространство признаков НС. Стоит отметить, что авторами рассматривается глубокая нейронная сеть со сверточными и/или рекуррентными слоями – одна из архитектур, которая более подробно будет рассмотрена в разделе выбора архитектуры ГНС на рис. 2.

4. Отнесение вектора признаков к одному из классов $\mathbf{f}_i : \tilde{\mathbf{x}} \to \mathbf{p}_{\tilde{\mathbf{x}}}$, где $\mathbf{p}_{\tilde{\mathbf{x}}}$ – вектор, содержащий вероятности классификации.

Выбор архитектуры ГНС. Сверточные нейронные сети особенно хорошо подходят для решения задач распознавания образов благодаря их способности к свертыванию параметров, извлечению признаков из локальных входных шаблонов и получению эффективных и модульных представлений данных. Однако все эти свойства могут быть применены и для обработки временных последовательностей, поскольку время можно рассматривать как пространственное измерение, подобно высоте или ширине двумерного изображения [13]. Одномерные сверточные слои могут распознать локальные шаблоны в последовательности [11]. Данная способность одномерных сверточных сетей может быть использована при обработке временных рядов сетевого трафика для классификации состояний сети, в том числе определения его нормального состояния и состояний с аномальными активностями.

Однако сверточные сети не чувствительны к порядку следования временных интервалов, поскольку в отличие от рекуррентных сетей лишены памяти [11]. «Классические» слои рекуррентной сети обрабатывают последовательность путем перебора ее элементов и сохранения состояний, полученных при обработке предыдущих элементов. Но зачастую обработка следующего элемента последовательности требует информацию не о предыдущем элементе, а об элементе/элементах из более отдаленного прошлого. «Классические» рекуррентные сети отчасти способны решать и данную за-

дачу, но для этого требуется увеличение количества рекуррентных слоев. И здесь возникает проблема затухания градиента: по мере увеличения количества слоев сеть в конечном итоге становится необучаемой [11]. Для решения данной проблемы имеются иные подходы: слой LSTM (Long Short-Term Memory) на основе алгоритма долгой краткосрочной памяти [2, 14] и слой GRU (Gated Recurrent Unit) [14, 15].

Стратегия, которой придерживаются некоторые авторы [10], – объединение скорости и легкости сверточных сетей с чувствительностью к порядку рекуррентных сетей: использование сверточной сети для предварительной обработки данных перед передачей их в рекуррентную сеть. Этот прием способен оправдать себя наибольшей степени, когда последовательность включает несколько тысяч интервалов и более, т.е. не реальна для обработки рекуррентной сетью. Сверточная сеть превратит длинную последовательность в более короткую последовательность высокоуровневых признаков, после чего они будут поданы на вход рекуррентной сети (рис. 1) [11]. Согласование этих слоев представлено в алгоритме (рис. 1).

Рис.1. Объединение одномерной сверточной и рекуррентной сетей.

Выберем в качестве ГНС следующую архитектуру (рис. 2):

1. Вход сети, представленный в виде трехмерного тензора, преобразованного из двумерного, поступает на вход слоя сворачивания, который преобразует вектор последовательности размера $C \times N \times S$ в вектор последовательности размером $C \times (N * S)$. Выходами слоя является карта выходных признаков на основе преобразованного входа, которая подается на вход сверточного слоя, а также минимальный размер пакета обработки, который подается на вход слоя разворачивания.

2. Два последовательных блока фильтров (с изменением размеров и количества фильтров):

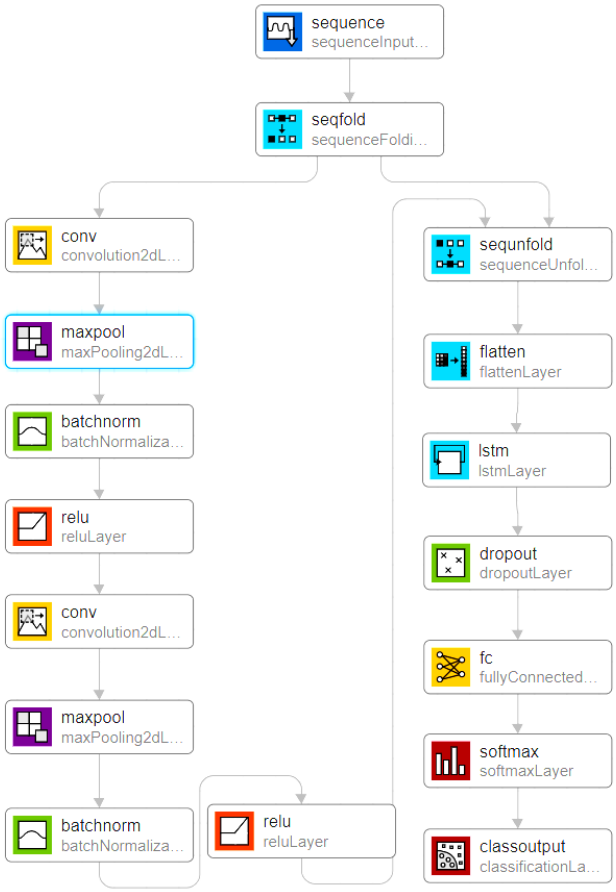


Рис.2. Архитектура нейронной сети в виде блоков посредством инструмента Deep Network Designer системы MATLAB.

2а. Слой свертки ядром $\mathbf{K}$ в слое l с линейной функцией активации σ^{linear}:

$$\mathbf{h}_{conv}^{l} = \sigma^{linear}(\mathbf{h}^{l-1} \cdot \mathbf{K} + b_{conv}^{l}), \quad (1)$$

где $\mathbf{h}_{conv}^{l}$ – выход слоя свертки; $\mathbf{h}^{l-1}$ – выход предыдущего слоя; b_{conv}^{l} – коэффициент сдвига.

2б. Слой нормализации:

$$\mathbf{h}_{bn}^{l} = a \cdot \frac{\mathbf{h}^{l-1} - \frac{1}{m}\sum_{i=1}^{m}\mathbf{h}^{l-1}}{\sqrt{\frac{1}{m}\sum_{i=1}^{m}(\mathbf{h}^{l-1} - \frac{1}{m}\sum_{i=1}^{m}\mathbf{h}^{l-1}) + e}} + b, \quad (2)$$

где $\mathbf{h}^{l-1}$– выход предыдущего слоя; m – количество параметров, обработанных предыдущем слоем; b – коэффициент сдвига; a – коэффициент масштабирования; e – константа для численной стабильности.

2в. Слой активации ReLU:

$$\mathbf{h}_{activ}^{l} = \sigma^{\mathrm{Re}LU}(\mathbf{h}^{l-1}). \quad (3)$$

Функция ReLU – линейная ректификация, или отсечение отрицательной части скалярной величины $\sigma^{\mathrm{Re}LU}(\mathbf{z}) = \max(0, \mathbf{z})$.

2г. Слой пуллинга (pooling, англ. – группировка): взятие максимального (среднего) значения из обрабатываемого блока информации, т.е. обобщение информации от предыдущего слоя: $h^l_{polling} = \max(\mathbf{h}^{l-1})$.

3. Слой разворачивания осуществляет операции, обратные слою сворачивания: преобразования вектора последовательности размера $C \times (N * S)$ в последовательность размером $C \times N \times S$.

4. Слой сглаживания, преобразующий пространственные размеры в один канал.

5. Рекуррентный слой LSTM. На вход слоя поступает «укороченная» последовательность высокоуровневых признаков, полученных от предыдущего слоя свертки.

LSTM-сеть использует специальные механизмы: фильтр забывания (*f*), входной (*i)* и выходной (*o*) фильтры. В отличие от рекуррентного блока, который просто вычисляет взвешенную сумму входных сигналов и использует нелинейную функцию активации, каждый *j*-й LSTM-блок поддерживает карту памяти c_t^j в момент времени *t*. Выход h_t^j LSTM-блока вычисляется следующим образом:

$$h_t^j = o_t^j \tanh(c_t^j), \quad (4)$$

где tanh – обозначение функции гиперболического тангенса; o_t^j – выходной фильтр, модулирующий объем содержимого памяти и вычисляемый следующим образом:

$$o_t^j = \sigma(W_o \mathbf{x}_t + U_o \mathbf{h}_{t-1} + b_o)^j, \quad (5)$$

здесь $\mathbf{x}$ – входная последовательность; σ – функция активации; U – обучаемый параметр; b – смещение.

Ячейки памяти c_t^j обновляются путем частичного забывания существующей памяти и добавления новой памяти $\tilde{c}_t^j$:

$$c_t^j = f_t^j c_{t-1}^j + i_t^j \tilde{c}_t^j, \ \tilde{c}_t^j = \tanh(W_c \mathbf{x}_t + U_c \mathbf{h}_{t-1})^j. \quad (6)$$

Степень забывания существующей памяти модулируется фильтром забывания f_t^j. Степень добавления нового содержимого в ячейку памяти модулируется входным фильтром i_t^j. Фильтры вычисляются следующим образом:

$$f_t^j = \sigma(W_f \mathbf{x}_t + U_f \mathbf{h}_{t-1} + b_f)^j, \ \ i_t^j = \sigma(W_i \mathbf{x}_t + U_i \mathbf{h}_{t-1} + b_i)^j. \quad (7)$$

6. Слой отсева шумов, или выпадающий слой, в котором все элементы предыдущего слоя становятся равными 0 с вероятностью между 0 и 100% (по умолчанию 50%), используется для предотвращения переобучения.

7. Чтобы спрогнозировать метки классов, сеть заканчивается полностью связанным слоем с количеством нейронов, равным количеству классов, слоем *softmax*, который определяет вероятности принадлежности последовательности классу, и выходным слоем классификации, который в соответствии с выходом слоя *Softmax* устанавливает соответствующие метки класса. *Softmax* преобразует выход предыдущего слоя в новый вектор той же размерности, где каждая координата полученного вектора будет числом в интервале [0, 1] и будет трактоваться как вероятность того, что соответствующий объект принадлежит классу, а сумма координат равна 1.

Результаты моделирования

Для экспериментального моделирования использовалась общедоступная база данных KDD [16]. База содержит почти 5 миллионов (4 891 469) классифицированных по 22 типам экземпляров сетевого состояния на основании 41 признака. Все признаки информативно не равнозначны. В соответствии с [17] только в 12 признаках содержится 99% информации для классификации, а согласно [18] для обнаружения и классификации 9 из 22 атак достаточно 29 признаков, однако это обстоятельство не учитывалось при получении указанных ниже результатов. Исходная база данных содержит данные как числового, так и символьного типа.

Для обработки все признаки символьного вида были преобразованы в числовые.

В базе атаки делятся на четыре категории:

DOS – отказ в обслуживании: back, land, Neptune, pod, smurf, teardrop;

U2R – атаки, направленные на получение зарегистрированным пользователем привилегий сетевого администратора: buffer_overflo, loadmodule, perl, rootkit;

R2L – атаки, направленные на получение доступа незарегистрированного пользователя: ftp_write, guess_passwd, imap, multihop, phf, spy, warezclient, warezmaster;

Probe – сканирование сетевых портов: ipsweep, nmap, portsweep, satan.

При анализе количества эталонов для каждой из атак, содержащейся в БД, можно сделать вывод, что для использования нейросетевых технологий для обучающей выборки могут быть использованы только эталоны нормального состояния сети и 5 атак: ipsweep (12 481 – количество эталонов), Neptune (1 072 917), portsweep (10 413), satan (15 892), smurf (2 807 886). Для получения описанных ниже результатов использовались эталоны трех состояний сети: нормальное состояние и две атаки: Neptune, smurf, так как именно

эталоны этих состояний составляют более 99,2% всей БД. Таким образом, для моделирования использовалась база данных трех состояний сети, содержащая матрицу *4853584×42*, где каждая запись определяет состояние компьютерной сети по 41-му признаку, а последний столбец – номер класса.

LSTM-сеть. Рассматривались различные варианты сочетаний объема обучающей и тестовой выборок. Моделирование производилось в среде MATLAB 2019 на персональном компьютере со следующими параметрами Intel Core i7-4710HQ, 4 ядра по 2,5 GHz, ОЗУ 6 Гб, ГПУ GeForce GTX 850M, 64-разрядная операционная система Windows. Была использована стандартная LSTM-сеть (рис. 3).

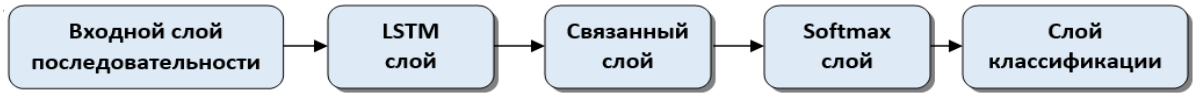


Рис. 3. Структура LSTM-сети.

Результаты моделирования представлены в табл. 1 и на рис. 4.

Таблица 1

№	Размер обучающей выборки	Размер тестовой выборки	Размер преобразованного тензора	Точность на обучающей выборке, %	Точность на тестовой выборке, %	Время обучения, мин.
1	3 000 x 41	3 000 x 41	150 x 20 x 41	100	100	0,7
2	12 000 x 41	3 000 x 41	600 x 20 x 41	100	100	1,5
3	120 000 x 41	30 000 x 41	6 000 x 20 x 41	100	100	16
4	1 200 000 x 41	300 000 x 41	60 000 x 20 x 41	100	99,29	75
5	300 000 x 41	75 000 x 41	600 x 500 x 41	100	100	1,05

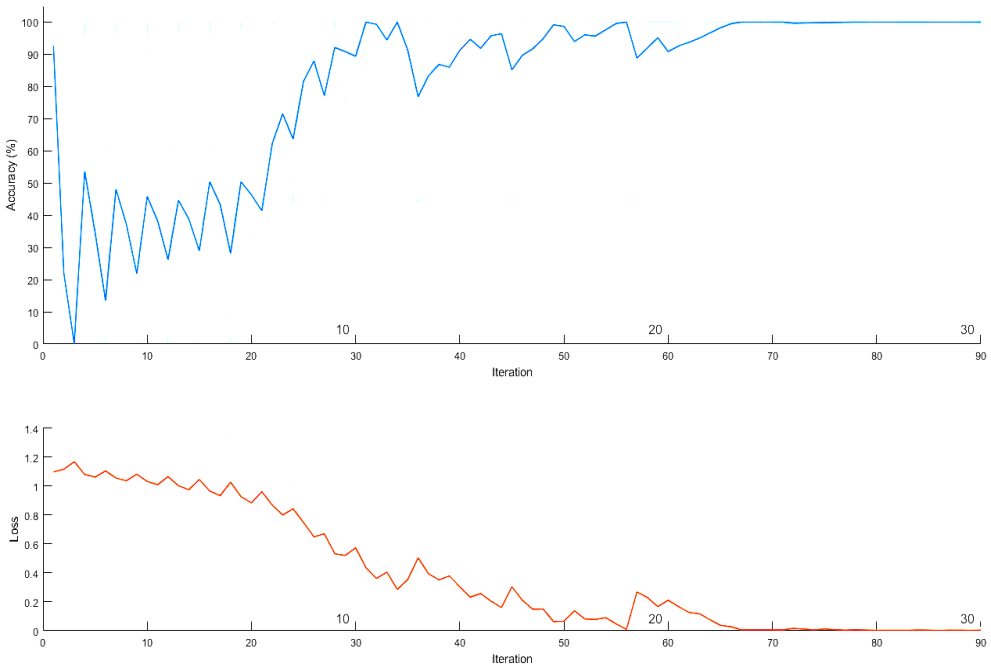


Рис. 4. Графическое отображение результатов обучения для эксперимента № 3: точность и значение потерь.

В экспериментах в LSTM-слое использовалось 100 нейронов. Для оптимизации машинных затрат и получения результата за оптимальное время информационные блоки из двумерного тензора преобразовывались в трехмерные. Без этого преобразования время обучения увеличивается на порядок, – например, для эксперимента № 3 оно превысило 6 часов.

ГНС со сверточными и рекуррентными слоями. Моделировалась ГНС на основе набора сверточных слоев и рекуррентного LSTM-слоя, архитектура которой представлена на рис. 2. Моделирование производилось в системе MATLAB 2019 на персональном компьютере со следующими параметрами: Intel Core i7-4710HQ, 4 ядра по 2,5 GHz, ОЗУ 6 Гб, ГПУ GeForce GTX 850M, 64-разрядная операционная система Windows. В экспериментах в LSTM-слое использовалось 100 нейронов. Результаты моделирования представлены в табл. 2 и на рис. 5.

Таблица 2

№	Размер обучающей выборки	Размер тестовой выборки	Размер преобразованного тензора	Точность на обучающей выборке, %	Точность на тестовой выборке, %	Время обучения, мин.
1	150 000 x 41	30 000 x 41	300 x 500 x 41	100	100	7,25
2	300 000 x 41	75 000 x 41	3000 x 100 x 41	100	99,8	13,45
3	900 000 x 41	225 000 x 41	1 800 x 500 x 41	100	98,67	35,1
4	300 000 x 41	75 000 x 41	600 x 500 x 41	100	100	8,1

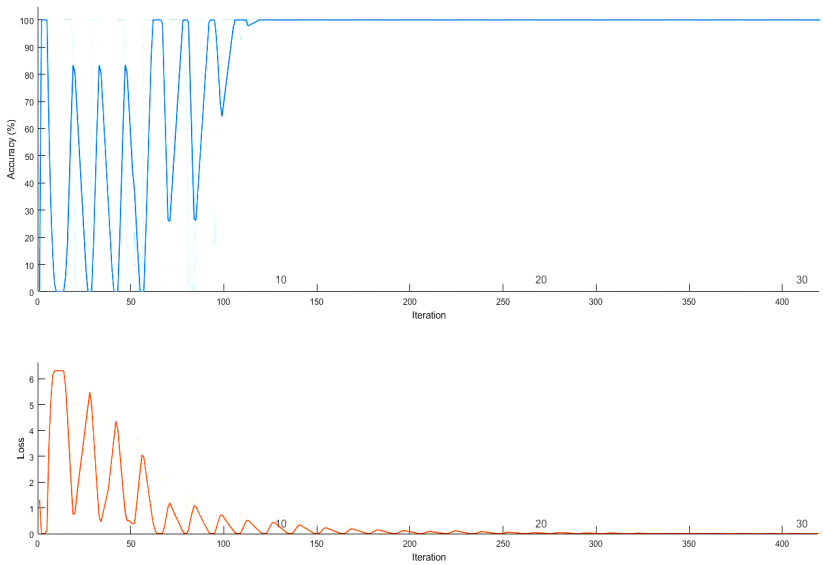
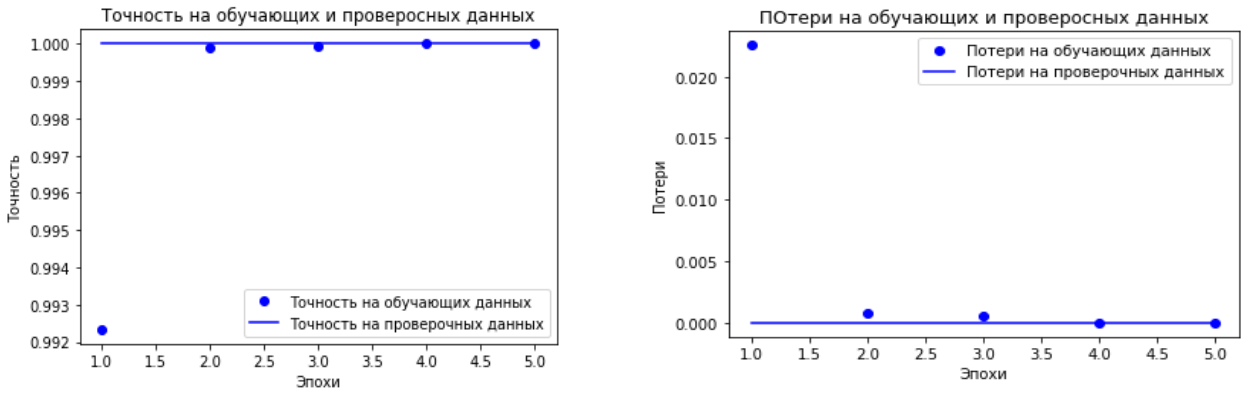


Рис. 5. Графическое отображение результатов обучения для эксперимента № 2: точность и значение потерь.

Для оптимизации машинных затрат и получения результата за более оптимальное время информационные блоки из двумерного тензора преобразовывались в трехмерные. Без данного преобразования время обучения увеличивается на порядок, – например, для эксперимента с размером обучающей выборки 45 000 x 41 оно составило 83 мин. (рис. 6).



а) изменения точности б) изменение потерь

Рис. 6. Графическое отображение результатов обучения для эксперимента.

Как видно из таблиц, применение дополнительного сверточного слоя увеличивает время обучения в целом, не увеличивая точности. Это можно объяснить тем, что по сути предъявленная для обучения последовательность не является длинной и содержит только 41 признак (и тысячи примеров), что с хорошей точностью обрабатывается рекуррентной сетью без дополнительной сверточной обработки. Однако в реальных условиях данные для классификатора с сетевых устройств компьютерной сети будут поступать непрерывно, включая тысячи интервалов, поэтому стратегия совместного использования сверточный и рекуррентных слоев, по мнению авторов, в реальных условиях может оказаться более оправданной.

Оптимизация параметров ГНС на основе генетического алгоритма. Качество и время обучения нейронной сети напрямую зависят от ее структуры, т.е. от количества слоев, нейронов в слоях, весовых коэффициентов, параметров функций активаций и прочее. Порой слишком громоздкая и/или не оптимальная структура сети позволяет достичь тех же (или худших) показателей, чем оптимальная структура, однако при этом затрачивается значительное количество вычислительных и временных ресурсов, что является критерием неэффективной работы. По этой причине поиск оптимальных параметров нейронной сети для решения конкретной прикладной проблемы остается актуальной задачей, и желательно, чтобы поиск таких параметров осуществлялся автоматически. Сверточные нейронные сети не являются исключением. В данном разделе для оптимизации параметров ГНС используется генетический алгоритм.

Генетические алгоритмы моделируют процессы природной эволюции и относятся к эволюционным методам поиска. С помощью генетического алгоритма можно получить решение, соответствующее глобальному оптимуму или близкое к нему [10]. В их основе лежит основная концепция эволюции – выживание или отбор наилучших индивидов, скрещивание и мутация. Генетический алгоритм представляет собой следующий объект:

$$ГА(P^0, r, l, sl, Fu, cr, m, ot), \quad (8)$$

где $ГА$ – генетический алгоритм; P^0 – исходная популяция; r – количество элементов популяции; l – длина битовой строки кодирующей решение; sl – оператор селекции; Fu – функция фитнесса (функция полезности) определяющая «пригодность» решения; cr – оператор кроссинговера, определяющий возможность получения нового решения; m – оператор мутации; ot – оператор отбора [19]. Одна из схем генетического алгоритма с элементами самоконфигурации представлена на рис. 7.

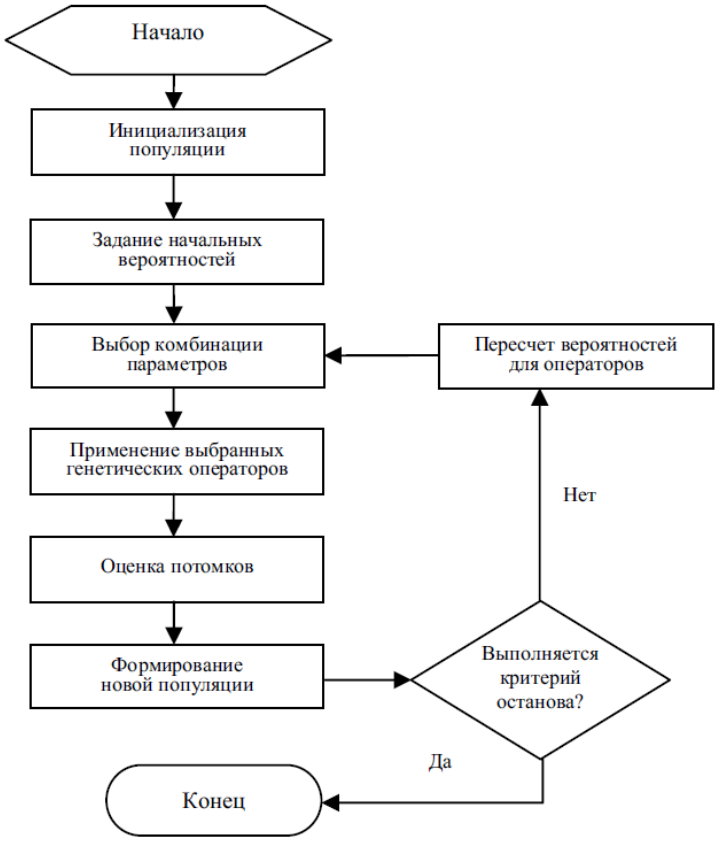


Рис. 7. Блок-схема генетического алгоритма.

Генетический алгоритм использовался для оптимизации следующих параметров ГНС со сверточными и рекуррентными слоями (рис. 2): количество нейронов в LSTM-слое, размер блока в слое пуллинга, значение вероятности в слое отсеивания шумов, количество фильтров и размеры этих фильтров в сверточных слоях.

Ввиду того, что по своей сути генетический алгоритм осуществляет поиск оптимальных параметров путем случайного подбора, комбинирования и вариации искомых параметров, подбор структуры нейронной сети на основе генетического алгоритма является процедурой, требующей значительных вычислительных мощностей. В связи с этим в качестве параметров генетического алгоритма использовались следующие: число поколений – не более 30,

число популяции – не более 50. Моделирование производилось в системе MATLAB 2019 на персональном компьютере со следующими параметрами: Intel Core i7-4710HQ, 4 ядра по 2,5 GHz, ОЗУ 6 Гб, ГПУ GeForce GTX 850M, 64-разрядная операционная система Windows. В зависимости от исходных параметров работа генетического алгоритма занимала от 1-2 до 12 часов. В табл. 3 представлены некоторые значения найденных при поиске генетическим алгоритмом параметров ГНС. На рис. 8 показано изменение значения штрафной функции при работе генетического алгоритма.

Таблица 3

№	Количество нейронов в LSTM-слое	Размер блока в слое пулинга	Значение вероятности в слое отсеивания шумов,%	Количество фильтров в сверточном слое	Размер фильтра в сверточном слое	Точность, %
1	40	5	89	10	10	33
2	70	5	74	7	6	66
3	100	5	52	9	4	67
4	60	4	94	10	4	67
5	50	7	49	4	5	33
6	60	5	87	6	4	67
7	80	6	36	9	6	66
8	40	9	2	5	6	33
9	50	7	49	4	5	100
10	50	4	25	6	6	100

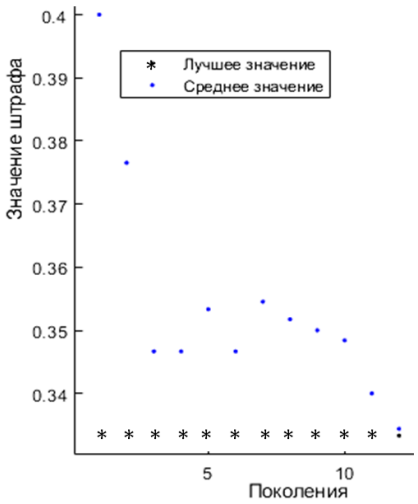


Рис. 8. Поиск решения генетическим алгоритмом.

При использовании генетического алгоритма для оптимизации параметров ГНС было отмечено следующее (что можно заметить из табл. 3):

1. Лучшей точности удавалось достигать при среднем числе нейронов.

2. Вероятность в слое отсеивания должна быть не больше 50%.

3. Значения, определяющие размер фильтра и их количество, должны быть соразмерными.

Таким образом, при наличии доступных вычислительных и временных

ресурсов использование генетических алгоритмов для оптимизации параметров ГНС позволит на этапе обучения получить более качественную модель, которая теоретически уже в режиме online позволит моделировать данные, максимально отражающие реальную картину происходящего, что в свою очередь позволит управляющему аппарату принимать верные и своевременные решения. В качестве альтернативы генетическому алгоритму авторы рассматривают возможность использования муравьиных алгоритмов [20] или роевый интеллект в целом [15].

Заключение

В работе представлена постановка задачи многоклассовой сетевой классификации компьютерных атак. Рассмотрена возможность использования технологии глубоких нейронных сетей для решения указанной задачи.

Выбрана архитектура глубокой нейронной сети на основе стратегии объединения набора сверточных и рекуррентных LSTM-слоев. Предложена оптимизация параметров выбранной глубокой нейронной сети на основе генетического алгоритма.

Представлены результаты моделирования, которые показывают возможность решения задачи сетевой классификации за приемлемое время и с ограниченными машинными ресурсами.

ЛИТЕРАТУРА

1. *LeCun Y., Bottou L., Bengio Y., Haffner P.* Gradient Based Learning Applied to Document Recognition // IEEE Intelligent Signal Processing. – 1998. – P. 306–351.
2. *Hochreiter S., Schmidhuber J.* Long-Short Term Memory // Neural Computation. – 1997. – Vol. 9, No. 8. – P. 1735–1780.
3. *Амосов О.С., Баена С.Г., Иванов Ю.С., Хтайк Со*. Система автоматического управления шлагбаумом с использованием технологий нечеткого вывода и компьютерного зрения // Интернет-журнал «Науковедение». – 2017. – Т. 9, № 1. – С. 42.
4. *Амосов О.С., Амосова С.Г., Жиганов С.В., Иванов Ю.С., Пащенко Ф.Ф.* Вычислительный метод распознавания ситуаций и объектов в кадрах непрерывного видеопотока с использованием глубоких нейронных сетей для систем контроля и управления доступом // Известия РАН. Теория и системы управления. – 2020. – №5. – С. 71-87.
5. *Амосов О.С., Амосова Л.Н., Магола Д.С.* Оценивание случайных последовательностей с использованием регрессии и вейвлетов // Информатика и системы управления. – 2009. – № 3 (21). – С. 101-109.
6. *Амосов О.С., Баена С.Г.* Иерархический подход построения интеллектуальной информационно-телекоммуникационной системы безопасности вуза // Международная конференция по мягким вычислениям и измерениям. – 2017. – Т. 1. – С. 188-191.
7. *Amosov O. S. and Baena S. G.* Decomposition Synthetic Approach for Optimum Nonlinear Estimation // IFAC – Papers OnLine. – 2015. – Vol. 48(11). – P. 819-824.

8. *Amosov O. S., Ivanov Y. S., and Zhiganov S. V.* Human localization in the video stream using the algorithm based on growing neural gas and fuzzy inference // XII Intelligent Systems Symposium, INTELS'16, Procedia Computer Science. – 2017. – No 103. – P. 403-409.
9. *Mesarovich M.D. and Takakhara Y.*, General Systems Theory: Mathematical Foundations, Academic Press New York, San Francisco, London, 1975.
10. Методы классической и современной теории автоматического управлениюю Учебник в 5 т. – Изд. 2-е, перераб. и доп. – Т.5: Методы современной теории автоматического управления / под ред. К.А. Пупкова, Н.Д. Егупова. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2004.

Шолле Ф. Глубокое обучение на Python. – СПб.: Питер, 2018.

11. *Amosov O. S., Amosova S. G., Ivanov Y. S., Zhiganov S. V.* Using the deep neural networks for normal and abnormal situation recognition in the automatic access monitoring and control system of vehicles // Neural Computing and Applications. – 2020.
12. *Mallat S.* A wavelet tour of signal processing. – Academic press, 1999.
13. *Будыльский Д. В.* GRU и LSTM: современные рекуррентные нейронные сети // Молодой ученый. – 2015. – №15. – С. 51-54.
14. *Chung J., Gulcehre C., Cho K.H., Bengio Y.* Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. – 2014. – arXiv:1412.3555.
15. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html. (дата обращения 10.09.2020).
16. *Марков Р.А., Бухтояров В.В., Попов А.М.* Исследование нейросетевых технологий для выявления инцидентов информационной безопасности // Молодой ученый. – 2015. – № 23. – С. 55-60.
17. *Комар М.П.* Нейросетевой метод идентификации компьютерных атак // Оптико-электронные информационно-энергетические технологии. – 2010. – № 2. – С. 105-109.
18. The MathWorks, Inc. (ЦИТМ «Экспонента»). Документация MATLAB. https://docs.exponenta.ru/gads/ga.html. (дата обращения 10.09.2020)
19. *Штовба С.Д.* Муравьиные алгоритмы // Exponenta Pro. Математика в приложениях. – 2003. – №4. – С.70-75.
20. *Beni G., Wang J.* Swarm Intelligence in Cellular Robotic Systems // Proceed. NATO Advanced Work-shop on Robots and Biological Systems. – Tuscany, Italy, 1989, June 26-30.

E-mail:
Амосов Олег Семенович – osa18@yandex.ru;
Амосова Светлана Геннадьевна – amosovasg@yandex.ru;
Магола Дмитрий Степанович – dmagola@list.ru.