С.В. БЕЗОБРАЗОВ, В.А. ГОЛОВКО
Брестский государственный технический университет, Республика Беларусь
bescase@gmail.com, gva@bstu.by

АЛГОРИТМЫ ИСКУССТВЕННЫХ ИММУННЫХ СИСТЕМ И НЕЙРОННЫХ СЕТЕЙ ДЛЯ ОБНАРУЖЕНИЯ ВРЕДОНОСНЫХ ПРОГРАММ

В работе представлены архитектура нейросетевого иммунного детектора, входящего в состав нейросетевой искусственной иммунной системы для обнаружения вредоносных программ, а также алгоритмы его обучения и функционирования.

1. Алгоритмы построения и функционирования нейросетевой искусственной иммунной системы для обнаружения вредоносных программ

Рассмотрим процессы генерации, обучения, отбора и функционирования иммунных детекторов на основе нейронных сетей. Представим нейросетевой иммунный детектор (НИД) в виде черного ящика, который имеет n-входов и два выхода (рис. 1).

Рис. 1. Нейросетевой иммунный детектор

Выходные значения детектора формируются после подачи всех образов на него в соответствии со следующим выражением:

$$Z_1 = \begin{cases} 1, & \textit{если чистый файл} \\ 0, & \textit{иначе.} \end{cases}$$
$$Z_2 = \begin{cases} 1, & \textit{если вирус} \\ 0, & \textit{иначе.} \end{cases} \qquad (1)$$

Обучающая выборка формируется из чистых файлов (класс чистых программ) и вредоносных программ (класс вредоносных программ). Желательно также иметь представителей всех типов вредоносных про-

грамм – черви, троянские программы, макровирусы и т.д. [5]. При обучении нейронной сети [6] указываем, где данные из чистых файлов, а где из вредоносных программ.

Пусть T – множество чистых файлов, а F – множество вредоносных файлов. Из них случайным образом формируется множество входных образов для обучения i-го детектора.

$$X_i = \begin{bmatrix} X_i^1 \\ X_i^2 \\ \dots \\ X_i^L \end{bmatrix} = \begin{bmatrix} X_{i1}^1 & X_{i2}^1 & \dots & X_{in}^1 \\ X_{i1}^2 & X_{i2}^2 & \dots & X_{in}^2 \\ \dots & & & \\ X_{i1}^L & X_{i2}^L & \dots & X_{in}^L \end{bmatrix}, \quad (2)$$

где L – размерность обучающей выборки.

Соответственно, множество эталонных образов выглядят следующим образом:

$$l_i = \begin{bmatrix} l_i^1 \\ l_i^2 \\ \dots \\ l_i^L \end{bmatrix} = \begin{bmatrix} l_{i1}^1 & l_{i2}^1 \\ l_{i1}^2 & l_{i2}^2 \\ \dots & \\ l_{i1}^L & l_{i2}^L \end{bmatrix}. \quad (3)$$

Эталонные выходные значения для i-го детектора формируются следующим образом:

$$l_{i1}^k = \begin{cases} 1, & \text{если } X_i^k \in T \\ 0, & \text{иначе.} \end{cases} \quad (4)$$

$$l_{i2}^k = \begin{cases} 1, & \text{если } X_i^k \in F \\ 0, & \text{иначе.} \end{cases}$$

Обучение каждого детектора осуществляется с целью минимизации суммарной квадратичной ошибки детектора. Суммарная квадратичная ошибка i-го детектора определяется следующим образом:

$$E_i = \frac{1}{2} \sum_{k=1}^{L} \sum_{j=1}^{2} (Z_{ij}^k - l_{ij}^k)^2, \quad (5)$$

где Z_{ij}^{k} – значение j-го выхода i-го детектора при подаче на вход его k-го образа.

Общий алгоритм функционирования нейросетевой иммунной системы [1] можно представить в виде следующей последовательности:

1. Генерация начальной популяции иммунных детекторов, каждый из которых представляет собой искусственную нейронную сеть со случайными синаптическими связями:

$$D = \left\{ D_i, \quad i = \overline{1,r} \right\}, \tag{6}$$

где D_i – i-й нейросетевой иммунный детектор, r – общее количество детекторов.

2. Обучение сформированных иммунных нейросетевых детекторов. Обучающая выборка формируется случайным образом из совокупности чистых файлов (как правило, это разнообразные системные утилиты операционной системы), и из совокупности вредоносных программ, или их сигнатур. Эталонные выходные значения нейронной сети формируются согласно (4).

3. Отбор (селекция) нейросетевых иммунных детекторов на тестовой выборке. На данной итерации уничтожаются те детекторы, которые оказались неспособны к обучению, и детекторы, в работе которых наблюдаются различные недостатки (например, ложные срабатывания). Для этого каждый детектор проверяется на тестовой выборке. В результате для каждого детектора определяется значение квадратичной ошибки E_i (5).

Селекция детектора производится следующим образом:

$$D_i = \begin{cases} 0, & \text{если } E_i \neq 0 \\ D_i, & \text{иначе.} \end{cases} \tag{7}$$

где 0 обозначает операцию уничтожения детектора.

4. Каждый детектор наделяется временем жизни и случайным образом выбирает файл для сканирования из совокупности файлов, которые он не проверял.

5. Сканирование каждым детектором выбранного файла, в результате которого определяются выходные значения детекторов Z_{i1}, Z_{i2}, i=1,…,r.

6. Если i-й детектор не обнаружил вирус в сканируемом файле, т.е. Z_{i1}=1 и Z_{i2}=0, то он выбирает следующий файл для сканирования. Если время жизни i-го детектора закончилось, то он уничтожается и вместо него генерируется новый детектор.

7. Если i-й детектор обнаружил вирус в сканируемом файле, т.е. Z_{i1}=0 и Z_{i2}=1, то подается сигнал об обнаружении вредоносного файла и осуществляются операции клонирования и мутации соответствующего детектора. Операция мутации заключается в дополнительном обучении детекторов-клонов на обнаруженном вредоносном файле. В результате создается совокупность детекторов, настроенных на обнаруженную вредоносную программу

$$D_i = (D_{i1}, D_{i2}, \ldots, D_{in}). \qquad (8)$$

8. Отбор клонированных детекторов, которые являются наиболее приспособленными к обнаружению вредоносной программы. Если $E_{ij} < E_i$, то детектор прошел отбор. Здесь E_{ij} – суммарная квадратичная ошибка j-го клона i-го детектора, которая вычисляется на вредоносном файле.

9. Детекторы-клоны осуществляют сканирование файлового пространства компьютерной системы до тех пор, пока не произойдет уничтожение всех проявлений вредоносной программы.

10. Формирование детекторов иммунной памяти. На этой итерации определяются нейросетевые иммунные детекторы, показавшие наилучшие результаты при обнаружении присутствующего в компьютерной системе вируса. Детекторы иммунной памяти находятся в системе достаточно длительное время и обеспечивают защиту от повторного заражения.

Особенностью предложенного алгоритма является то, что каждый нейросетевой иммунный детектор является полностью самостоятельным объектом. Он случайным образом выбирает файл из списка для его проверки. После проверки одного файла детектор переходит к следующему случайно выбранному файлу. При этом соблюдается принцип децентрализации системы безопасности, построенной на основе комбинации методов нейронных сетей и искусственных иммунных систем, что значительно повышает отказоустойчивость и защищенность системы в целом.

2. Структура и алгоритм обучения нейросетевого иммунного детектора

В процессе циркуляции НИД происходит их непрерывная эволюция путем уничтожения старых и формирования новых детекторов [1]. После генерации новых детекторов происходит процесс их обучения, трудоемкость которого пропорциональна размерности обучающей выборки. Поэтому для увеличения быстродействия нейросетевой искусственной иммунной системы необходимо выбрать такой класс нейронной сети, кото-

рый характеризуется минимальным размером обучающей выборки. Рассмотрим многослойный персептрон [6, 7], который состоит из n нейронов распределительного слоя, m нейронов скрытого слоя и 2 нейронов выходного слоя. Общее количество настраиваемых параметров (весовых коэффициентов и пороговых значений) в такой сети определяется следующим образом:

$$V = m \cdot (n + 3) + 2. \tag{9}$$

Для хорошей классификации размер обучающей выборки должен определяться в соответствии со следующим выражением [7]:

$$L \approx \frac{V}{\varepsilon}, \tag{10}$$

где ε – допустимая точность классификации.

Пусть $n = 128$, $m = 10$ и $\varepsilon = 0,1$. Тогда $L \approx 13120$. Аналогичный результат можно получить для мультирекуррентных нейронных сетей [10, 11].

Рассмотрим аналогичную сеть встречного распространения [6, 7] с идентичным количеством нейронных элементов в слоях. В скрытом слое будем использовать нейронные элементы Кохонена. В этом случае нет жестких требований к размерности обучающей выборки. Достаточно, чтобы размер обучающей выборки был следующим:

$$L \geq 2 \cdot m. \tag{11}$$

Поэтому выберем в качестве основы нейросетевого иммунного детектора нейронную сеть встречного распространения.

На рис. 2 изображена архитектура нейросетевого иммунного детектора, который состоит из трех слоев нейронных элементов и арбитра.

На вход такого детектора в режиме функционирования подаются фрагменты проверяемого файла, которые формируются в соответствии с методом скользящего окна. Первый слой нейронных элементов является распределительным. Он распределяет входные сигналы на нейронные элементы второго (скрытого) слоя. Количество нейронных элементов распределительного слоя равняется размерности скользящего окна. Второй слой состоит из нейронов Кохонена, которые используют конкурентный принцип обучения и функционирования в соответствии с правилом «победитель берет все» [6, 7]. Третий слой состоит из двух линейных нейронных элементов, которые используют линейную функцию активации. Арбитр осуществляет процедуру окончательного решения о принадлежности сканируемого файла к вирусному или чистому классу.

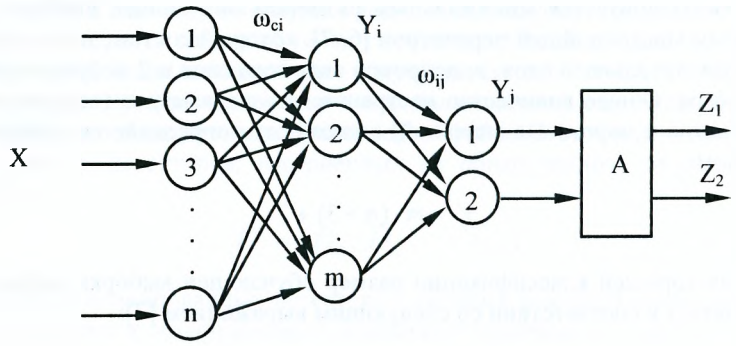

Рис. 2. Нейросетевой иммунный детектор

Рассмотрим выбор количества нейронов в слое Кохонена. Нейронный слой Кохонена осуществляет кластеризацию входного пространства образов, в результате чего образуются кластеры различных образов, каждому из которых соответствует свой нейронный элемент. Количество нейронов слоя Кохонена равняется m. Причем

$$m = p + r, \tag{12}$$

где p – количество первых нейронов слоя Кохонена, которые соответствуют классу чистых программ, r – количество последних нейронов слоя Кохонена, активность которых характеризует класс вредоносных программ.

При обучении нейросетевых иммунных детекторов используется обучающая выборка, состоящая из 80% образов чистого класса и из 20% образов вредоносного класса. Данное соотношение классов (1:4) было получено экспериментальным путем и показало наилучшие результаты [4].

Алгоритм формирования обучающей выборки состоит в следующем:

1) Формируется совокупность чистых и вирусных файлов.

2) Из сформированной выборки случайным образом выбираются k чистых и h вредоносных файлов.

3) Из каждого файла случайным образом выбираются A фрагментов длиной n, в результате чего образуется обучающая выборка размерностью $L = (k+h)\cdot A$.

Для обучения нейронов слоя Кохонена используется контролируемое конкурентное обучение [6, 7]. При таком обучении весовые коэффициенты нейрона победителя модифицируются только тогда, когда происходит корректная классификация входного образа, т.е. входной образ соответст-

вует заданному множеству нейронов в слое Кохонена. Так как в слое Кохонена используется p нейронов для чистых входных образов и r нейронов для вредоносных входных образов, то корректная классификация происходит, если при подаче на вход сети чистого фрагмента победителем является один из первых p нейронов слоя Кохонена. Аналогичным образом корректная классификация происходит, если при подаче на вход сети вирусного фрагмента победителем является один из r последних нейронов слоя Кохонена. В остальных случаях происходит некорректная классификация.

Пусть P и J характеризуют соответственно чистый и вредоносный файл. Тогда правило корректной классификации можно представить в виде следующей импликации:

$$\begin{aligned} P \wedge k = 1,2 \ldots p \rightarrow T, \\ J \wedge k = p+1, r \rightarrow T, \end{aligned} \tag{13}$$

где T обозначает корректную классификацию.

При корректной классификации весовые коэффициенты нейрона-победителя усиливаются

$$\omega_{ck}(t+1) = \omega_{ck}(t) + \gamma(X_c - \omega_{ck}(t)), \tag{14}$$

а при некорректной классификации ослабляются:

$$\omega_{ck}(t+1) = \omega_{ck}(t) - \gamma(X_c - \omega_{ck}(t)), \tag{15}$$

где γ – шаг обучения.

Алгоритм обучения слоя Кохонена состоит из следующих шагов:

1. Случайная инициализация весовых коэффициентов нейронов слоя Кохонена.

2. Подается входной образ из обучающей выборки на нейронную сеть и производятся следующие вычисления:

а) вычисляется Евклидово расстояние между входным образом и весовыми векторами нейронных элементов слоя Кохонена

$$D_i = |X - \omega_i| = \sqrt{(X_1 - \omega_{1i})^2 + (X_2 - \omega_{2i})^2 + (X_n - \omega_{ni})^2}, \tag{16}$$

где $i = \overline{1, m}$.

b) определяется нейронный элемент-победитель с номером k

$$D_k = \min_j D_j. \tag{17}$$

c) производится модификация весовых коэффициентов нейрона-победителя в соответствии с выражением (14), если при подаче на вход сети чистого фрагмента победителем является один из первых p нейронов или при подаче на вход сети вредоносного фрагмента победителем является один из r последних нейронов сети Кохонена. В противном случае производится модификация весовых коэффициентов нейрона-победителя в соответствии с выражением (15).

3. Процесс повторяется, начиная с пункта 2, для всех входных образов.

4. Обучение производится до желаемой степени согласования между входными и весовыми векторами, т.е. до тех пор, пока значение суммарной квадратичной ошибки не станет равной величине заданного порога.

Третий слой, состоящий из двух линейных нейронных элементов, осуществляет отображение кластеров, сформированных слоем Кохонена, в два класса, которые характеризуют чистые и вирусные входные образы. В общем случае выходное значение j-го нейрона третьего слоя определяется следующим образом:

$$Y_j = \sum_{i=1}^{m} \omega_{ij} \cdot Y_i, \tag{18}$$

где ω_{ij} – весовой коэффициент между i-м нейроном слоя Кохонена и j-м нейроном линейного слоя.

Если нейрон-победитель в слое Кохонена имеет номер k, то выходное значение j-го нейрона третьего слоя равняется

$$Y_j = \omega_{kj} \cdot Y_k. \tag{19}$$

Для соответствующего отображения входных образов в два класса матрица весовых коэффициентов третьего слоя должна формироваться следующим образом:

$$\omega_{kj} = \begin{cases} 1, & \text{если } k = 1,2 \ldots p \text{ и } j = 1 \text{ или } k = p+1 \ldots r \text{ и } j = 2 \\ 0, & \text{если } k = 1,2 \ldots p \text{ и } j = 2 \text{ или } k = p+1 \ldots r \text{ и } j = 1. \end{cases} \tag{20}$$

Так, например, для p=8 и r=2, получается следующая матрица весовых коэффициентов:

$$W^T = \begin{bmatrix} 1\,1\,1\,1\,1\,1\,1\,1\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,1\,1 \end{bmatrix}. \tag{21}$$

Арбитр принимает окончательное решение о том, является ли сканируемый файл вредоносным. Для этого он вычисляет количество чистых и вредоносных фрагментов сканируемого файла в соответствии со следующими выражениями:

$$\overline{Y_1} = \sum_{k=1}^{L} Y_1^k, \tag{22}$$

$$\overline{Y_2} = L - \overline{Y_1} = \sum_{k=1}^{L} Y_2^k, \tag{23}$$

где L – множество образов сканируемого файла, Y_i^k – выходное значение i-го нейрона линейного слоя при подаче на вход сети k-го образа.

Далее определяются вероятности принадлежности сканируемого файла соответственно к чистому и вредоносному классу:

$$P_T = \frac{\overline{Y_1}}{L} \cdot 100\%, \tag{24}$$

$$P_F = 1 - P_T = \frac{\overline{Y_2}}{L} \cdot 100\%. \tag{25}$$

Окончательное решение о принадлежности файла к чистому классу арбитр принимает следующим образом:

$$Z_1 = \begin{cases} 1, & \text{если } P_T > 80\% \\ 0, & \text{иначе.} \end{cases} \tag{26}$$

Соответственно, решение о принадлежности сканируемого файла к вредоносному классу принимается в соответствии со следующим выражением:

$$Z_2 = \begin{cases} 1, & \text{если } P_F > 20\% \\ 0, & \text{иначе.} \end{cases} \tag{27}$$

Пространство выходных значений арбитра представлено в табл. 1. Если выходные значения арбитра имеют нулевые значения, то сканируемый

файл отправляется на дополнительную проверку другому нейросетевому иммунному детектору.

Таблица 1

Пространство выходных значений арбитра

Z_1	Z_2	класс
1	0	чистый
0	1	вредоносный
0	0	не определено

3. Алгоритм функционирования нейросетевого иммунного детектора

В процессе сканирования проверяемого файла на нейросетевой детектор последовательно подаются фрагменты файла по методу скользящего окна.

Алгоритм функционирования нейросетевого иммунного детектора в режиме сканирования файла можно свести к следующей последовательности шагов:

1. Устанавливаются следующие начальные значения:

$$\overline{\overline{Y_1}}(k-1)=0, \qquad (28)$$
$$\overline{\overline{Y_2}}(k-1)=0.$$

2. По методу скользящего окна последовательно подаются входные образы (k=1,L) из сканируемого файла на нейронную сеть и для каждого входного образа производятся следующие вычисления:

a) Определяется Евклидово расстояние между входным образом и весовыми векторами нейронов слоя Кохонена (формула (16)).

b) Определяется нейронный элемент-победитель с номером k (формула (17)).

c) Вычисляются выходные значения линейных нейронных элементов третьего слоя (формула (19)).

d) Определяется количество чистых и вредоносных фрагментов сканируемого файла:

$$\overline{Y_1}(k)=\overline{Y_1}(k-1)+Y_1^k, \qquad (29)$$
$$\overline{Y_2}(k)=\overline{Y_2}(k-1)+Y_2^k. \qquad (30)$$

3. Вычисляются вероятности принадлежности сканируемого файла соответственно к чистому и вредоносному классу (формулы (24) и (25) соответственно).

4. На основании вычислений вероятностей принимается решение о принадлежности сканируемого файла к одному из классов, в соответствии с выражениями (26) и (27).

5. Если Z_1=0 и Z_2=0, то назначается другой нейросетевой иммунный детектор для повторной проверки файла.

4. Результаты исследований

В табл. 2 представлены результаты сравнительного анализа обнаружения вредоносных программ различными антивирусными продуктами. Для теста были выбраны следующие антивирусные продукты: Антивирус Касперского [8] с актуальными вирусными базами; Антивирус Касперского с устаревшими вирусными базами; антивирусный продукт NOD 32 [9] с отключенными вирусными базами, но с задействованным эвристическим анализатором; и разработанная нами нейросетевая искусственная иммунная система. В таблице «*OK*» означает решение антивирусной программы о том, что файл является чистым.

Как видно из полученных результатов, антивирус с актуальными вирусными базами обнаружил все вредоносные программы, которые использовались в эксперименте. Это объясняется тем, что в антивирусных базах содержались сигнатуры используемых в эксперименте вредоносных программ. Антивирус с устаревшими базами обнаружил только половину присутствующих вирусов, что наглядно отражает неспособность сигнатурного метода обнаруживать неизвестные вредоносные программы. Антивирус NOD 32, который использовал эвристический анализатор, обнаружил только семь вирусов, что является очень низким показателем для надежной современной системы безопасности и отражает проблемную ситуацию обнаружения неизвестных вредоносных программ с помощью эвристических методов. Искусственная иммунная система показала наилучшие результаты.

Один нейросетевой иммунный детектор способен обнаруживать несколько вредоносных программ. Причем детектор приобретает способность обнаруживать принципиально новые вредоносные программы.

Рассмотрим приближенную оценку вероятности обнаружения вредоносных программ нейросетевой искусственной иммунной системой.

Пусть P_i – вероятность обнаружения вредоносного файла i-м детектором. При сканировании файлового пространства r независимыми детекторами вероятность необнаружения вредоносной программы определяется следующим образом:

$$g(r) = \prod_{i=1}^{r}(1 - P_i). \tag{31}$$

Отсюда можно получить следующее выражение для оценки вероятности обнаружения вредоносной программы нейросетевой иммунной системой:

$$P(r) = 1 - \prod_{i=1}^{r}(1 - P_i). \tag{32}$$

Таблица 2

Результаты сравнительного анализа обнаружения

Имя файла	Антивирус Касперского (актуал. базы)	Антивирус Касперского (устар. базы)	NOD32 (эвристическ. анализатор)	ИИС (на основе 4-х детекторов)
Backdoor.Win32.Agent.lw	Backdoor	OK	OK	OK
Backdoor.Win32.Agobot	Backdoor	Backdoor	Win32/Agobot	Вирус
Email-Worm.BAT.Maddas	Email-Worm	Email-Worm	OK	Вирус
Email-Worm.JS.Gigger	Email-Worm	Email-Worm	OK	Вирус
Email-Worm.VBS.Loding	Email-Worm	Email-Worm	OK	Вирус
Email-Worm.Win32.Zafi.d	Email-Worm	OK	NewHeur_PE	Вирус
Net-Worm.Win32.Bozori.a	Net-Worm	OK	Win32/Bozori	Вирус
Net-Worm.Win32.Mytob.a	Net-Worm	OK	Win32/Mytob	Вирус
Trojan-Downl.JS.Psyme.y	Trojan	OK	OK	Вирус
Trojan-Downl.Win32.Bagle	Trojan	OK	Win32/Bagle	Вирус
Trojan-Proxy.Daemonize	Trojan	Trojan	OK	OK
Trojan-Proxy.Mitglieder	Trojan	Trojan	Win32/Trojan	Вирус
Trojan-Proxy.Win32.Agent	Trojan	Trojan	OK	Вирус
Trojan-PSW.LdPinch	Trojan	Trojan	Win32/PSW	Вирус
Virus.Win32.Gpcode.ac	Virus.Win32	OK	OK	Вирус
Exploit.Win32.DebPloit	Exploit	OK	OK	OK

Как следует из последнего выражения, с увеличением количества детекторов увеличивается вероятность обнаружения вредоносной программы. Рассмотрим приближенную оценку количества детекторов для заданной достоверности обнаружения вредоносной программы $P(r)$. Предпо-

ложим, что вероятность обнаружения вредоносной программы каждым детектором приблизительно одна и та же и равняется P_0. Тогда

$$P(r) = 1 - (1 - P_0)^r. \quad (33)$$

Следовательно, для заданной вероятности обнаружения вредоносных программ $P(r)$ можно получить приближенную оценку количества детекторов в иммунной системе:

$$r = \frac{\ln(1 - P(r))}{\ln(1 - P_0)}. \quad (34)$$

Отсюда следует, что с увеличением популяции нейросетевых иммунных детекторов вероятность обнаружения вредоносных программ возрастает.

Рассмотрим теоретическую и экспериментальную оценки вероятности обнаружения вредоносной программы в зависимости от количества детекторов. Экспериментальная оценка такова:

$$P(k) = \frac{n(k)}{n}, \quad (35)$$

где k=1,r, r – общее количество детекторов, $n(k)$ – количество вредоносных программ, обнаруживаемых k-детекторами, n – общее количество вредоносных программ.

Для получения приближенной теоретической оценки необходимо определить вероятность обнаружения детектором вредоносной программы. Предположим, что вероятность обнаружения вредоносной программы каждым детектором является приблизительно одинаковой и равняется P_0. Тогда, согласно выражению (33)

$$P_0 = 1 - (1 - P(r))^{1/r}. \quad (36)$$

Пусть r = 24, тогда $P(r)$ = 0,78. Отсюда P_0 = 0,0325. Тогда теоретическая оценка вероятности обнаружения вредоносной программы в зависимости от количества детекторов определяется следующим образом:

$$P(k) = 1 - (1 - P_o)^k. \quad (37)$$

На рис. 3 приведены результаты экспериментальной и теоретической вероятности обнаружения вредоносной программы в зависимости от количества детекторов. Как следует из рисунка, теоретическая вероятность

хорошо аппроксимирует экспериментальную вероятность обнаружения вредоносных программ.

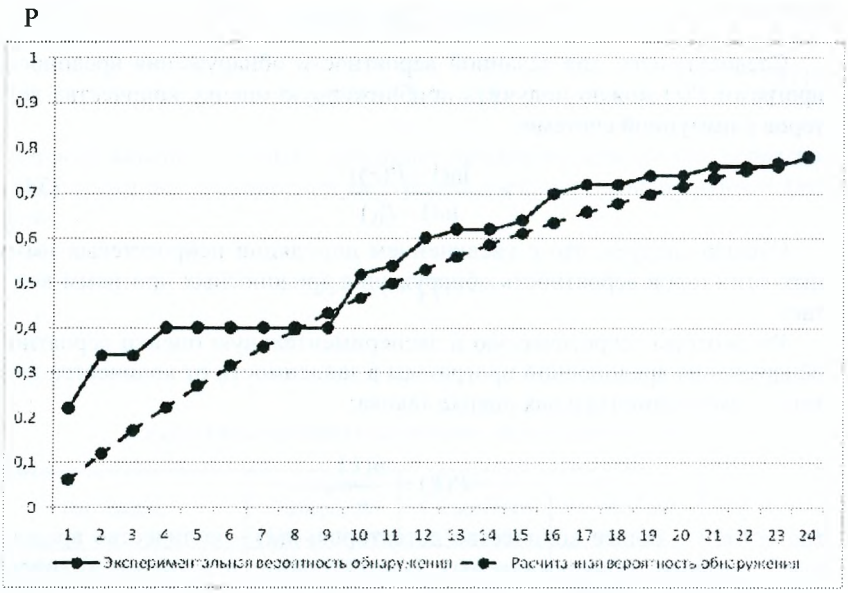

Рис. 3. Вероятности обнаружения вредоносных программ детекторами

Выводы

Разработана структура нейросетевого иммунного детектора для обнаружения вредоносных программ, которая состоит из трех слоев нейронных элементов и арбитра. Она характеризуется малым объемом обучающей выборки и отношением количеств нейронов чистого и вредоносного классов в слое Кохонена как 4:1. Предложенный нейросетевой иммунный детектор способен обнаруживать неизвестные вредоносные программы.

Список литературы

1. Безобразов С.В. Искусственные иммунные системы для защиты информации: применение LVQ сети // Нейроинформатика-2007: материалы IX Всеросс. науч.-техн. конф., Москва, МИФИ, 2007. С. 27-35.

2. Безобразов С.В., Головко В.А. Искусственные иммунные системы для защиты информации: обнаружение и классификация компьютерных вирусов // Нейроинформатика-2008: материалы IX Всеросс. науч.-техн. конф., Москва, МИФИ, 2008. – С. 23-27.

3. Bezobrazov S., Golovko V. Neural Networks for Artificial Immune Systems: LVQ for Detectors Construction // IDAACS'2007: proceedings of the 4 IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications. – Dortmund, 2007. – P. 180-184.
4. Bezobrazov S., Golovko V. Neural networks and artificial immune systems – malware detection tool // ICNNAI'2008: proceedings of the 5 International Conference on Neural Networks and Artificial Intelligence, Minsk, 27-30 May 2008. / Brest State University of Informatics and Radioelectronics. – Minsk, 2008. – P. 49-52.
5. Касперский Е. Компьютерное зловредство. СПб.: Питер, 2007. – 208 с.
6. Головко В.А. Нейронные сети: обучение, организация, применение // Нейрокомпьютеры и их применение : учеб. пособие. – М., 2001 – 256 с.
7. Хайкин С. Нейронные сети: полный курс. М.: Вильямс, 2006. – 1104 с.
8. http://www.kaspersky.ru.
9. www.esetnod32.ru.

А.В. ЛУКАНИН
Южно-Уральский государственный университет, Челябинск
ice_lc@mail.ru

ИСКУССТВЕННАЯ НЕЙРОННАЯ СЕТЬ ДЛЯ ГЕНЕРАЦИИ ФОРМ ПРОШЕДШЕГО ВРЕМЕНИ В СОВРЕМЕННОМ АНГЛИЙСКОМ ЯЗЫКЕ

Предлагается использовать многослойный персептрон для ассоциации орфографической записи основ английских глаголов с аддитивными правилами, преобразующими их в формы прошедшего времени, с целью её использования в задачах автоматической обработки текстов. Сеть обучается большому лексикону правильных и неправильных глаголов, после чего тестируется её способность к обобщению на других глаголах. Приводятся рекомендации по улучшению процесса обучения.

Введение

Один из взглядов на язык, восходящий к работам Н. Хомского [1-2], поддерживаемый Дж. Фодором [3] и в данный момент развиваемый С. Пинкером [4-5], предполагает, что абстрактные правила играют ключевую роль в обработке естественного языка. Альтернативная позиция принадлежит так называемому коннекционистскому направлению [6], которое рассматривает все когнитивные процессы, включая обучение языку, как постепенную подстройку весовых коэффициентов связей простых вычислительных устройств – нейронов, совокупность которых составляют искусственные нейронные сети (ИНС) – модели нервной системы челове-