УДК 621.8

В.И. Литвиненко

ИММУННЫЙ КЛАССИФИКАТОР ДЛЯ РЕШЕНИЯ ЗАДАЧ БИНАРНОЙ КЛАССИФИКАЦИИ (ТЕОРЕТИЧЕСКИЕ ОСНОВЫ)

Введение

Существует несколько теорий [1,2,3] и математических моделей [4,5,6,7], объясняющих иммунологические явления. Существует также определенное число компьютерных моделей [8,9,10,11], для моделирования различный компонентов входящих в состав иммунной системы и полное ее поведение с биологической точки зрения. С другой стороны, естественная иммунная система является источником вдохновения для разработки информационных методологий для решения прикладных задач. В настоящее время исследователи используют лишь три иммунологических метода: теорию иммунной сети, механизм отрицательного отбора и принцип клонального отбора. В настоящей работе мы исследуем механизм отрицательного отбора. Алгоритм отрицательного отбора и его модификации, несмотря накажущююся свою простоту является мощным инструментом для решения в первую очередь таких задач, как защита компьютерных сетей [12], защита от компьютерных вирусов [13], обнаружение аномальных объектов на изображениях [14,], сегментации образов [15], для обнаружения аномалий в временных рядах данных. [16], мониторинг вычислительных процессов в операционной системе UNIX [17], диагностика отказов в авиации [18].

Суть механизма отрицательного отбора заключается в том, что иммунная система стремится обеспечить толерантность к “своим” клеткам и молекулам. Это развивает способность иммунной системы обнаруживать неизвестные антигены при отсутствии какой-либо реакции на свои клетки. В течение текущего поколения с помощью псевдослучайного процесса генетической перестановки создаются рецепторы Т-клеток. После этого, они подвергаются цензурированию в вилочковой железе (Тимусе), посредством процесса который называется “отрицательным отбором”. В Тимусе, Т-клетки, реагирующие против “своих” белков, разрушаются и только те из них, которые не связываются со “своими” белками, позволяется покинуть Тимус.

 ISSN 1562-9945

Эти созревшие Т-клетки после этого циркулируют по всему телу, чтобы осуществлять свои иммунологические функции и защитить в целом организм против чужеродных антигенов.

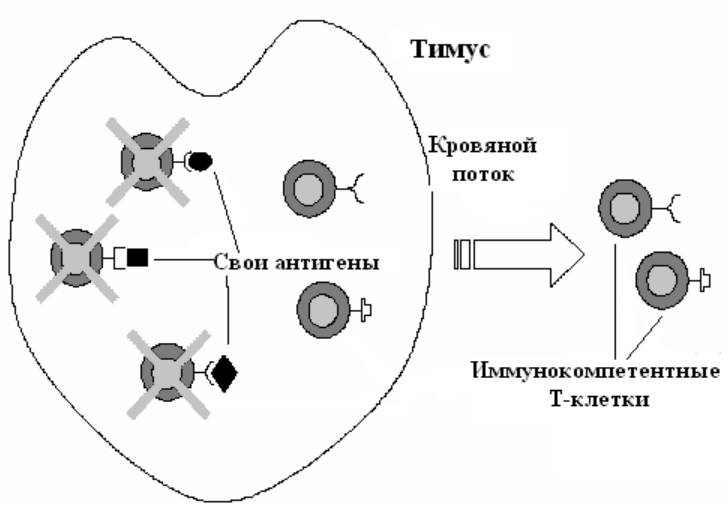

Рис.1 Упрощенное представление отрицательного отбора в Тимусе. Незрелые Т-клетки, которые распознают свои антигены, удаляются из репертуара. Наоборот, те, которые не распознают любой свой антиген, становятся иммунокомпетентными клетками и попадают в кровяной поток

Алгоритм *отрицательного отбора* (АОО) *был* предложен Стефанией Форрест в 1994 году [19], данный алгоритм применяется механизм, используемый иммунной системой при обучении Т-клеток для распознавания чужих антигенов и в дальнейшем препятствововании им распознавать собственные клетки организма как “свои”. Идея заключалась в генерировании набора бинарных датчиков создаваемых кандидатов и затем отказаться от тех из них, которые в дальнейшем распознают “свои” данные из набора обучающей выборки. В дальнейшем эти датчики или детекторы могут использоваться, для обнаружения каких-либо аномалий. Так что АОО состоит из трех стадий: определение “своих”, генерирование датчиков и контроль за возникновением аномалий. Были предложены различные разновидности этого алгоритма (датчик, генерирующий шаблоны). Первичными применениями АОО было в области обнаружение изменений (или аномалий), где датчики генерировались в дополнительное пространство, которое может обнаружить изменения в образцах данных. Главный компонент АОО – выбор соответствующего правила, которое определяет сходство между двумя образцами, чтобы осуществить классифицировацию свои/чужие (нормальные/ неправильные) образы.

Временная сложность генерирования датчиков линейна относительно размера "своих" данных. Необходима также оценка для определения размера набора датчиков чтобы гарантировать некоторый “уровень надежности” в обнаружении аномалий. Сгенерированные датчики имеющие высокий порог соответствия, становятся чувствительными к любой аномалии в данных, так что необходимо большое количество датчиков позволяющие достигнуть желательного уровня полной надежности. С другой стороны, если порог является слишком маленьким, не может быть возможной генерации разумного размера набора датчиков от доступных “своих”. Это предлагает, что значение порога может использоваться, чтобы настроить надежность обнаружения против риска ложных положительных распознаваний. Это особенность применяется при обнаружении изменений в общем случае.

В работе [20] приводятся результаты работы, где автор рассмотрел и сравнил пять схем поколения датчиков (бинарных) алгоритмов отрицательного отбора: исчерпывающий, линейный, “жадный”, бинарный шаблон, и мутация АОО. Последняя схема - измененная версия исчерпывающего алгоритма, который вводил соматичесукую гипермутацию и устранение избыточности к исчерпывающему АОО алгоритму для лучшей работы.

В работе [21] авторы проанализировали и сравнили различный правила бинарного соответствия в отрицательном отборе: r-смежное соответствие, соответствие r-куска, соответствие с использованием Хеммингового расстояния, и его разновидность – R&T-соответствие. Это таким образом обеспечивает директиву для того, чтобы выбрать различные правила соответствия для любых алгоритмов отрицательного отбора.

В других работах также исследовалось различное представление схемы отрицательного отбора и алгоритмов генерации датчиков для таких представлений [22]. В частности исследуемые представления включали гиперпрямоугольники (который можно интерпретировать как правила), нечеткие правила, и гиперсферы.

Предложено четыре различных алгоритма генерации датчиков, соответственно: 1) отрицательный отбор с правилами обнаружения (эволюционный алгоритм, для генерирования датчиков гиперкуба); 2) отрицательный отбор с нечеткими правилами обнаружения (эволюционный алгоритм, для генерирования датчиков нечетких правил), 3) от-

 ISSN 1562-9945

рицательный отбор на действительных числах (эвристический алгоритм, для генерирования гиперсферических детекторов), и 4) отрицательный отбор на основе рандомизированых действительных чисел (алгоритм позволяющий генерировать гиперсферические датчики основанный на методах Монте Карло). Разработан также гибридный алгоритм обучения, обладающий иммунитетом, который комбинирует способы генерирования детекторов 3) или 4) с алгоритмами классификации.

Другая разновидность модели отрицательного отбора предлагает использовать динамический алгоритм клонального отбора (Динамика), чтобы иметь дело с "чужим", задача заключается в обнаружения в условиях непрерывно меняющейся окружающей среде [23], В частности динамика основывается на идее Хофмеера [24] о динамике трех различных популяций: незрелая, зрелая, и популяция датчиков памяти. Первоначальные незрелые датчики, генерируются со случайными генотипами. Используя отрицательный отбор, новые незрелые датчики добавляются, для того чтобы общее количество датчиков сохранялось постоянным после предопределенного числа поколений (период поляризации Т). Если датчик - в пределах его предопределенной продолжительности жизни L, и соответствующее число является большим, чем предопределенный порог активации A, он становится датчиком памяти. Зрелые датчики используются, чтобы идентифицировать неизвестные нападения. Однако, необходимо подтверждение человека отвечающего за безопасность (ко-стимуляция, который делает этот подход, зависящим от человеческого взаимодействия. В отечественных исследованиях, к сожалению данный алгоритм все еще остается малоизвестным.

Постановка задачи. Целью предлагаемой работы является изложение методологии синтеза алгоритмов отрицательного отбора для решения задач классификации. На тестовых задачах нами проведены численные эксперименты по исследованию влияния некоторых параметров алгоритма отрицательного отбора на качество решения задач классификации.

АЛГОРИТМ ОТРИЦАТЕЛЬНОГО ОТБОРА

Фрмально алгоритм отрицательного отбора можно представить в виде:

$$NegA\lg = \left(C, P, M, r, n, s, pr\right) \qquad (1)$$

U – общее число детекторов кандидатов

S – множество детекторов определяемых как “свой”

M – множество детекторов определяемых как “чужой”

r – кросс-реактивный порог

n – общее число назначаемых детекторов

s – размер множества детекторов

pr – правило совпадения строк в r – смежных позициях

В обобщенном виде данный алгоритм представлен на рис.2:

1.Инициализация: генерация случайного начального множества детекторов-кандидатов (строки или вектора)

2.Старт алгоритма: пока множество детекторов заданного размера еще не сгенерировано, выполнить:

2.1. Оценка аффинности: определить аффинность между каждой “своей” клеткой и детектором-кандидатом.

2.2. Отбор: если детектор-кандидат распознает любой “свой” элемент, этот кандидат устраняется, в противном случае детектор-кандидат помещается во множество детекторов.

3.Мониторинг: после того как сгенерировано множество детекторов, изучается любое новое множество строк. В этом случае, если какой-либо элемент из множества детекторов распознал элемент изучаемого множества, то это означает, что был обнаружен “чужой” элемент.

Рис. 2 Обобщенный стандартный алгоритм отрицательного отбора

На этапе обучения система, используя обучающую выборку образов **(P)** (свои антигены), создает множество **(M)** комплиментарных **P** образов, называемых *детекторами*. Далее процесс распознавания происходит так: новый образ, поступающий в систему сравнивается с множеством детекторов и при отсутствии совпадений распознается как “свой”; при наличии совпадения исследуемого образа хотя бы с одним детектором, он распознается как “чужой”. Данную схему можно проиллюстрировать следующим образом (рис. 3, 4).

Более детально алгоритм отрицательного отбора можно описать следующим образом.

Стадия обучения.

 ISSN 1562-9945

1. Случайно генерируется множество кандидатов детекторов (**C**). Каждый элемент этого множества имеет тоже представление что и исследуемый образ.

2. Каждый элемент из множества **C** сравнивается с элементами из множества **P** и если обнаружено соответствие, то элемент множества **C** отбрасывается, иначе этот элемент сохраняется во множестве детекторов **M**.

Стадия распознавания.

Для исследуемой выборки **P*** сравнивается каждый ее элемент со всеми элементами множества детекторов **M**. Если обнаружено хотя бы одно совпадение, то исследуемый элемент из **P*** распознается как «чужой», иначе он считается «своим».

Выбор правила совпадения является критическим по отношению к производительности данного метода. Как правило в таких исследованиях используется правило совпадения “r-смежных битов”, позаимствованное из теоретической иммунологии. Согласно данному правилу, две строки совпадают друг с другом, если они идентичны по крайней мере в r-смежных позициях (где r-параметр правила совпадения).

Для сравнения элементов множеств между собой используется понятие степени совпадения элементов. Каждый элемент представлен в виде строки атрибутов, в качестве которых могут выступать двоичные числа, вещественные числа или просто символы. Для примера рассмотрим двоичное представление данных.

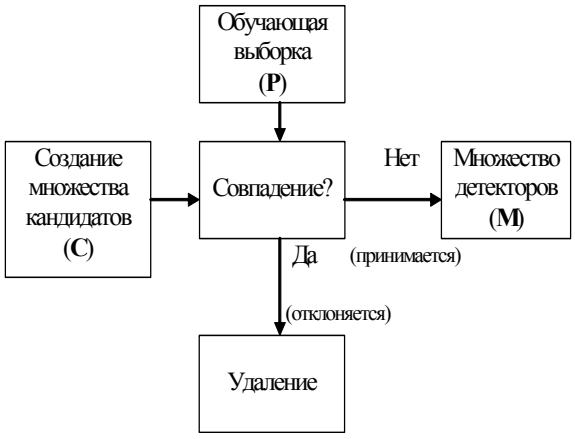

Рис. 3 Создание множества детекторов (обучение системы).

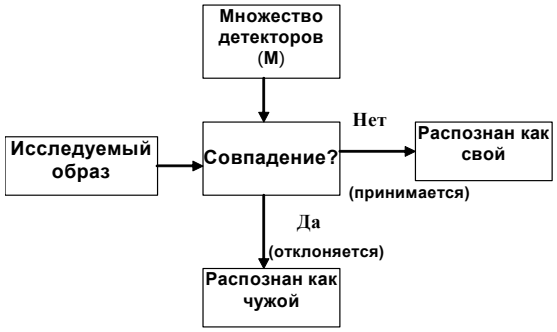

Рис.4 Работа системы (распознавание)

Образ на рисунке 3 может быть представлен в виде строки нулей и единиц таким образом, что черные пиксели означают «1», а белые «0».

Рис. 5 Входной образ для распознавания (20x20)

В этом случае степень совпадения может быть определена как Хеммингово расстояние между двумя двоичными строками:

$$D_H = \sum_{i=1}^{L} \delta_i \text{ , где } \quad \delta_i = \begin{cases} 1, & \text{если } p_i = m_i \\ 0, & \text{в противном случае} \end{cases}, \qquad (2)$$

где L – длина строки представления образа; p_i, m_i – значения соответствующих элементов сравниваемых строк.

Например следующие две строки имеют степень совпадения равную 10 (рис. 6).

1 0 0 0 1 1 1 1 0 1 0 1 1 0 0
0 0 0 1 1 0 0 1 0 0 0 1 1 0 0
• • • • • • • •

Рис. 6 Степень совпадения двух строк равная 10

Одним из параметров алгоритма является пороговое значение степени совпадения элементов множеств (φ), при превышении которого элементы считаются равными между собой.

В зависимости от того как представляются данные в системе, можно воспользоваться другими способами вычисления степени совпадения элементов множеств, например, Евклидовым расстоянием, Манхеттановским расстоянием и др.

Графическая интерпретация алгоритма отрицательного отбора

С целью более удобного представления рассмотрим двумерный случай распознавания, т.е. случай, когда распознаваемые строки (вектора) состоят из двух компонент, называемых атрибутами.

Пусть нам дано множество **U** – универсум и множество **S** векторов, которые классифицируются как “***свои***”, причем

 ISSN 1562-9945

S ⊂ U (рис.7)

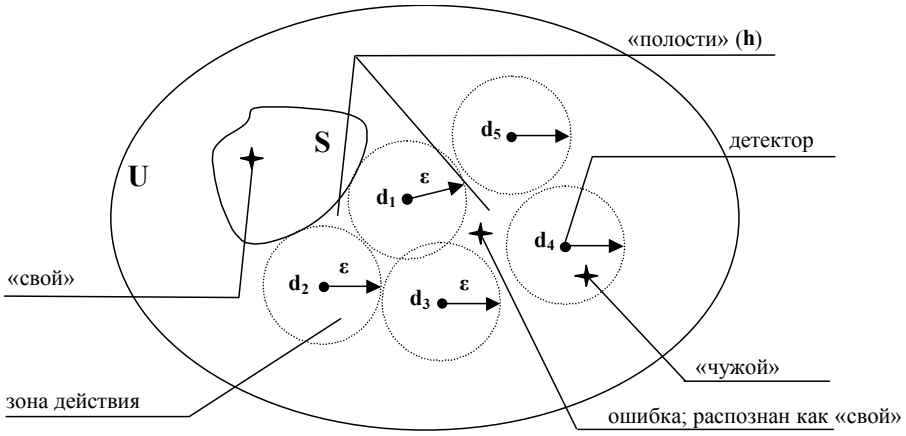

Рис. 7 Графическое представление объектов алгоритма

Алгоритм отрицательного отбора предполагает создание множества детекторов ***D*** таких, что $\forall d_i \in D \quad d_i \in U \wedge d_i \notin S$. Каждый детектор d_i имеет свою собственную окрестность распознавания или "зону действия". Размеры зоны действия детектора определяются параметрам $\sum$ (***Threshold***). Форма зоны определяется выбранным правилом сравнения строк (***Matching Rule***). Например, при выборе Евклидова расстояния в качестве правила сравнения, форма зоны действия детектора будет выглядеть, как показано на рисунке 8а. Рисунок 8б показывает зону действия в случае Манхеттеновского расстояния. Зоны действия детекторов могут пересекаться друг с другом, но обязательным условием является тот факт, что ни одна зона не должна пересекать границу множества ***S***. Это неизбежно приводит к образованию своеобразных "мертвых зон" или полостей ***h*** (рисунок 1).

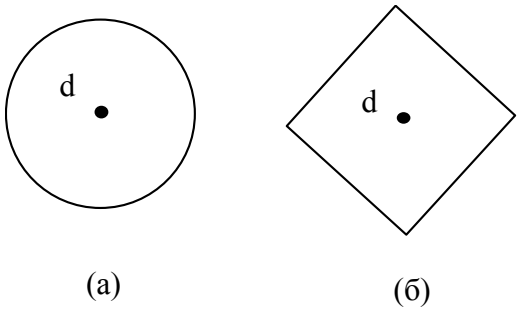

Рис. 8 Формы зон действия при различных правилах сравнения векторов (двумерный случай): а) Евклидово расстояние, б) Манхеттеновское расстояние.

При распознавании вектора проверяется его попадание в зону действия какого-либо детектора или нескольких детекторов. Если такое попадание обнаружено, то вектор распознается как "чужой" ина-

че он распознается как “свой”. При распознавании вектора попавшего в одну из полостей h_i, он будет ошибочно распознан как “свой”, так как полости не покрываются ни одним из детекторов. Таким образом, мы сталкиваемся с необходимостью выбора компромиссного решения. С одной стороны можно уменьшить размеры полостей уменьшая зоны действия детекторов, кроме того мы можем располагать детекторы плотнее друг к другу с большим количеством взаимных пересечений зон. Но и в том и в другом случае это приведет к увеличению количества детекторов, необходимых для сохранения максимального покрытия множества **U − S**, и следовательно к увеличению вычислительных затрат алгоритма. Выходом из данной ситуации может стать разработка алгоритма оптимизации покрытия детекторами множества **U − S**, а также разработка новых правил сравнения строк, генерирующих более сложные формы зон действия и одновременного использования в рамках одного алгоритма различных правил сравнения.

ПРОСТОЙ ПРИМЕР РАСПОЗНАВАНИЯ

В качестве самого простого примера мы выбрали распознавание двухкомпонентных векторов. Взаимное расположение классов векторов показано на рисунке 9.

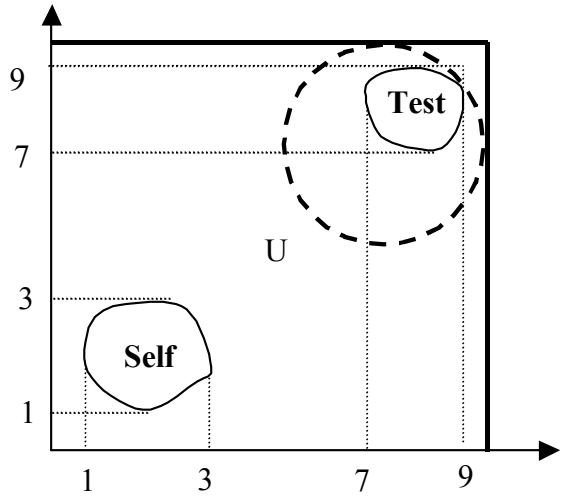

Рис. 9 Пример распознавания двухкомпонентных векторов.

В качестве обучающей выборки было выбрано 50 элементов множества, обозначенного на рисунке как “Self”. Тестовая выборка состояла из 10 элементов. Данный пример отличается удобством расположения класса “своих” векторов и класса тестируемых векторов. Де-

 ISSN 1562-9945

текторы практически всегда без труда полностью покрывают тестовый класс, что исключает возникновение ошибок распознавания.

ЗАДАЧА “IRIS FLOWER” И ПРОБЛЕМЫ РАСПОЗНАВАНИЯ

Другим, более сложным примером является известная задача распознавания сорта цветов Ириса по четырем характеристикам их соцветий. Три класса цветов Ириса (Setosa, Versicolor и Virginica) расположены таким образом, что возникают трудности при их разделении и, соответственно, распознавании (рис. 10).

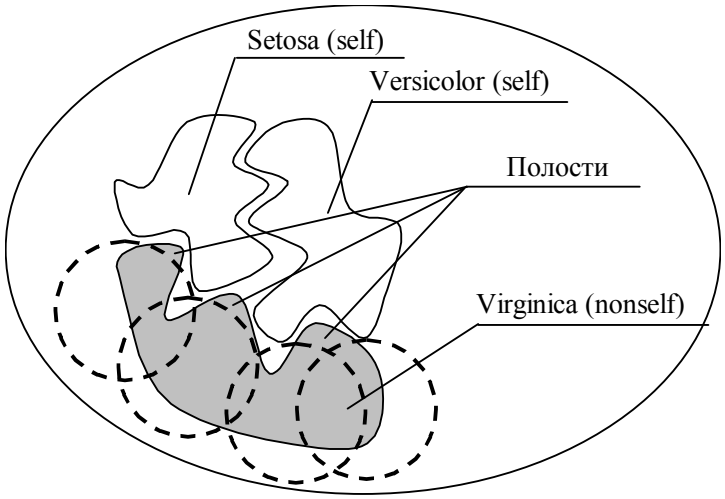

Рис. 10 Примерное взаимное расположение классов Setosa, Versicolor и Virginica. Для удобства представления вектора характеристик растений были спроецированы на двумерную плоскость.

В нашем примере мы объявили классы Setosa и Versicolor классами «своих» объектов, а класс Virginica – классом «чужих». Из рисунка видно, что при любом расположении детекторов остается невозможным полностью покрыть класс Virginica, что неизбежно приведет к ошибкам распознавания. Уменьшение радиуса зоны действия детекторов приведет к увеличению размера множества необходимых детекторов, а при достаточно высокой мощности универсума – к неоправданным затратам времени на их генерацию. Кроме того, увеличение количества детекторов замедлит и сам процесс распознавания. Ниже приведены возможные пути решения этой проблемы.

ПУТИ РЕШЕНИЯ ПРОБЛЕМЫ

Одним из компромиссных решений проблемы распознавания является использование динамических параметров при создании множества детекторов. В качестве таковых могут выступать параметр, определяющий размер зоны действия детектора ε (для Евклидова рас-

стояния это радиус зоны) и форма зоны действия (Matching Rule) (рис. 11).

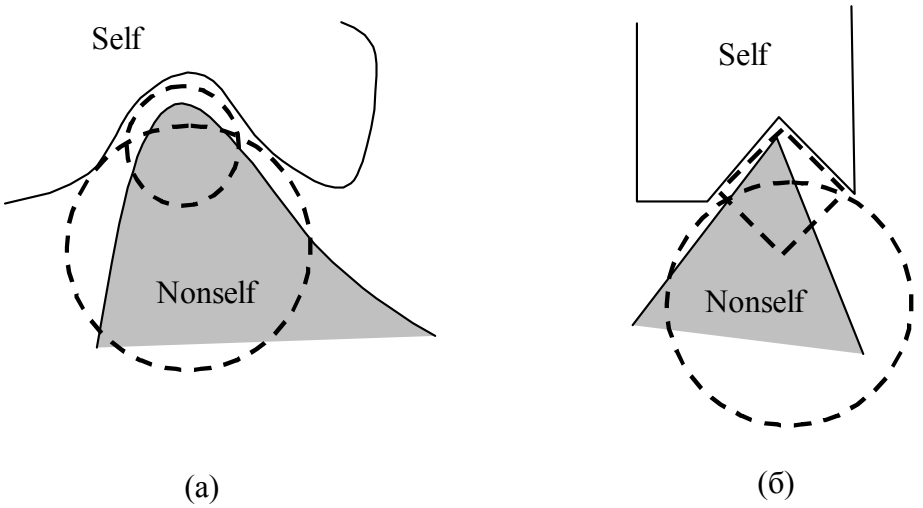

Рис. 11 Применение динамических параметров при генерации множества детекторов.

В этом случае пространственная форма детектора должна содержать в себе информацию о значениях параметров, которые он использует. Следовательно, представление детектора будет таким (рис. 12).

Детектор

Атрибут 1	Атрибут 2	Атрибут n	...	Threshold	Matching rule

Рис.12 Форма детектора при использовании динамических параметров

И, конечно нельзя забывать об усовершенствовании алгоритма генерации множества детекторов.

ТЕОРЕТИКО-МНОЖЕСТВЕННОЕ ОБОСНОВАНИЕ АЛГОРИТМА

На рисунке 13 показано множество строк и отношения между ними. Пространство строк U, и пространство детекторов U_d изображены отдельно друг от друга для ясности, хотя в данной работе $U = U_d$. *Вероятность совпадения* $P_м$: вероятность того, что случайно выбранная строка и детектор совпадут согласно правилу совпадения.*Вероятность ошибки* P_f: вероятность того, что отдельная случайно взятая “чужая” строка не совпадет ни с одним детектором из *R*.Обозначим N_X размер (количество элементов) множества X. В частности N_S – размер множества “своих” и N_R - размер множества детекторов.Имея множество “своих” строк $S \subset$ U, цель состоит в том,

 ISSN 1562-9945

чтобы найти множество детекторов $R \subset U_d$, которое совпадает с наибольшим количеством "чужих" строк из N ($N = U\text{-}S$), и при этом не совпадает ни с одной из "своих" строк из S. Детекторы в R выбираются из множества кандидатов детекторов C, и состоят из всех строк, которые не совпадают ни с одной строкой из S. Если бы мы включили все кандидаты детекторов в R, то мы бы смогли обнаружить все "чужие" строки во множестве N', которое может быть или может не быть равным N. Вообще, только небольшое подмножество кандидатов детекторов будет включено в R, позволяя нам обнаружить все "чужие" строки из D (при $D \subset$ N'(N). Вероятность ошибки $\boldsymbol{P_f = N_D/N_N}$, или $\boldsymbol{P_f}$ $\boldsymbol{N_D/N_U}$, если $\boldsymbol{N_S}$ P $\boldsymbol{N_U}$.

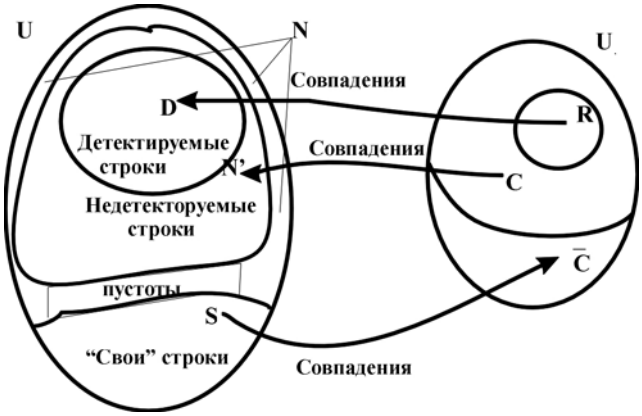

Рис. 13 – Отношения между подмножествами множества строк U и множества детекторов Ud. Множество строк разделено на "свои" строки S, и "чужие" строки N. Кандидатами детекторов ($\boldsymbol{C}$) – являются те строки, которые не соответствуют ни одной строке из S. Множество детекторов R – это подмножество кандидатов детекторов. "Чужие" строки далее подразделяются на обнаруживаемые "чужие" строки (N') и пустоты (H = N – N', на рисунке не помечены). Обнаруживаемые "чужие" строки могут быть обнаружены (множество D) или нет (множество F = N'- D, на рисунке не помечены), в зависимости от выбора множества R.

ОЦЕНКА ВЕРОЯТНОСТИ РАСПОЗНАВАНИЯ

Так как распознавание или обнаружение в алгоритме отрицательного отбора является по своей природе вероятностным процессом, мы должны уметь делать точные оценки на основе этих вероятностей для различных конфигураций решения задач распознавания и обнаружения.

Предположим, что мы имеем некоторую строку, которую мы хотим распознать. При этом строка может быть как прикладной про-

граммой, набором некоторых данных, или любым другим элементом компьютерной системы, который хранится в памяти. Используя алгоритм отрицательного отбора необходимо определить число и размер строк детекторов, которые будут гарантировать, например, что с определенной вероятностью будет обнаружено какое-либо изменение в строках.

Первоначально введем следующие обозначения:

N_{R_0} - Исходное количество строк-детекторов перед проверкой.

N_R - Число строк-детекторов после проверки.

N_S - Число “своих” строк.

P_M - Вероятность совпадения между двумя случайно выбранными строками.

f - Вероятность того что случайно выбранная строка не совпадет ни с одной из множества N_S “своих” строк $= (1 - P_M)^{N_S}$

P_f - Вероятность, что N_R детекторов не в состоянии осуществить распознавание.

Если P_M является маленьким а N_S велико, то

$$f \approx e^{-P_m N_S} \tag{1}$$

и,

$$N_R = N_{R_0} \times f \tag{2}$$

$$P_f = (1 - P_M)^{N_R} \tag{3}$$

Если P_M является маленьким, а N_R является большим, то

$$P_f \approx e^{-P_M N_R} \tag{4}$$

таким образом,

$$N_R = N_{R_0} \times f = \frac{-\ln P_f}{P_M} \tag{5}$$

Решая (5) относительно N_{R_0}, мы получаем следующее выражение:

 ISSN 1562-9945

$$N_{R_0} = \frac{-\ln P_f}{P_M \times (1 - P_M)^{N_S}} \quad (6)$$

Данная формула позволяет нам предсказывать число начальных строк (N_{R_0}), которые будут необходимы, чтобы обнаружить случайное изменение, как функцию вероятности обнаружения (1-P_f), число "своих" строк, (N_S), и правило совпадения (P_M). N_{R_0} – минимизируется путем выбора правила совпадения такого, как

$$P_M \approx \frac{1}{N_S} \quad (7)$$

Предшествующий анализ позволяет нам оценивать вычислительные затраты алгоритма следующим способом. Первоначально необходимо отметить, что метод базируется на двух основных действиях: (1) генерации случайных строк фиксированной длины и (2) сравнении двух строк, для того чтобы определить удовлетворяют ли они критерию совпадения (больше чем ***r*** смежных совпадений).

Предположим, что данные действия требуют постоянного времени. Тогда, временная сложность Фазы 1 будет пропорциональна количеству строк в R_0 (то есть, N_{R_0}) и числу строк в S (то есть, N_S). Уравнение, 6 оценивает N_{R_0} основываясь на размере распознаваемого множества данных (N_s), требуемой надежности обнаружения (P_f), и специфическом правиле совпадения (P_M). Затраты на полные проверки в Фазе II будут пропорциональны числу строк в R (то есть, N_R) и числу строк в S (то есть, N_S).

ВЫВОДЫ

Основываясь на вышеупомянутом анализе, мы можем сделать несколько замечаний об алгоритме отрицательного отбора:

1. Не требует априорных знаний.

2. Он является настраиваемым: мы можем выбрать желаемую вероятность обнаружения (P_f), и затем посчитать число требуемых строк детекторов, как функцию размера N_S (строк, которые будут распознаваться), используя выражения 3 и 7. Увеличение вероятности обнаружения увеличивает вычислительные расходы, т.к. увеличиваются размеры R_0 и R. Можно выбирать желаемую вероятность обнаружения в соответствии с необходимостью

3.N_R (*число строк-детекторов после проверки*) не зависит от N_S (*число "своих" строк*) при фиксированной P_M и P_f (Уравнение 5). То есть размер множества детекторов не обязательно растет с увеличением числа распознаваемых строк. Это подразумевает, что имеется возможность распознавать очень большие объемы данных с использованием маленького множества детекторов.

4. Вероятность обнаружения увеличивается по экспоненте с числом независимых алгоритмов обнаружения.

Если N_t = число копий алгоритма в системе, тогда вероятность ошибки обнаружения (распознавания) равна $P = (P_f)^{N_t}$ Данная особенность является основным преимуществом алгоритма. Например, одна копия алгоритма обнаружения, $N_t = 1$ с 46 детекторами может распознать множество данных любого размера с вероятностью 90,6%. Используя 10 различных копий алгоритма $N_t = 10$, такой же уровень распознавания может быть получен менее, чем при наличии 4-х детекторов на каждый алгоритм.

5. Обнаружение является симметричным. Изменения в множестве детекторов обнаружены тем же самым процессом совпадения, который обнаруживает изменения в "своих". Это означает, что, когда обнаружено изменение не существует никакого способа определить было ли изменение в своих или в детекторах. Преимущество состоит в том что "свой" осуществляет такое же распознавание детекторов, как и детекторы распознают "своих".

6. Схема распознавания является весьма распределенной. Небольшие сегменты защищаемого объекта могут проверяться по отдельности, различные участки могут иметь независимо созданные множества детекторов для одного и того же объекта, кроме того детекторы во множестве детекторов могут управляться независимо друг от друга. Нет необходимости во взаимодействии между детекторами и множествами детекторов до тех пор пока не будет обнаружено какое-либо изменение.

7. Распознавание является локальным. Классические методы распознавания изменений (такие, например, как методы контрольной суммы) обычно требуют, чтобы весь набор данных был проверен сразу. Данный метод позволяет проверять данные маленькими порциями и в случае, когда детектор действительно находит аномалию, она мо-

жет быть локализована в пределе одной строки, которую тестирует детектор.

ЛИТЕРАТУРА

1. N. K. Jerne. Towards a network theory of the immune system. Ann. Immunol. (Inst. Pasteur), 125C:373{389, 1974.
2. Ronald R. Mohler, Carlo Bruni, and Alberto Gandol_.A System Approach to Immunology. Proceedings of the IEEE, 68(8):964{990, 1980.
3. Gerard Weisbuch. A shape space approach to the dynamics of the immune system. Journal of Theoretical Biology, 143(4):507-522, 1990.
4. Alan S. Perelson. Immune network theory. Immunological Reviews, (10):5{36, 1989.
5. Franciso J. Varela and John Stewart. Dynamics of a class of immune networks I. Global Stability of idiotype interactions. Journal of Theoretical Biology, 144(1):93{101, 1990.
6. Марчук Г.И. Математические модели в иммунологии.–М.Наука, 1980.–256 с.
7. Белых Л.Н. Анализ математических моделей в иммунологии / под ред. Г.И. Марчука. – М.: Наука. Гл. ред. физ.-мат.лит., 1988.-192 с.
8. Franco Celada and Philip E. Seiden. A computer model of cellular interactions in the immune system. Immunology Today, 13(2):56{62, 1992.
9. Richard G. Weinand. Somatic mutation, a_nity maturation and antibody repertoire: A computer model. Journal of Theoretical Biology, 143(3):343{382, 1990.
10. Gerard Weisbuch. A shape space approach to the dynamics of the immune system. Journal of Theoretical Biology, 143(4):507{522, 1990.
11. Молер Р.Р., Бруни К., Гандолфи А. Системный подход в иммунологии // ТИИЭР, 1980, том 68, № 8, с.25-56.
12. P. D'haeseleer, S. Forrest, and P. Helman. An immunological approach to change detection: algorithms, analysis, and implications. In Proceedings of IEEE Symposium on Research in Security and Privacy, Oakland, CA, May 1996.pp.110-119.
13. Kephart, J.O., Sorkin, G.B., Swimmer, M.& White S.R. (1999), “Blueprint for a Computer Immune System” , In Artificial Immune

Systems and Ther Applications, D. Dasgupta (ed.), Springer-Verlag, pp.242-259

14. Aissu, H.& Mizutani H. (1996), "A Rule Acquisition for Image Processing Using Immune Mechanism", Proc. Of the 12th Fuzzy System Symposium, pp. 75-78
15. McCoy, D.F. & Devarajan, V. (1998). "Artificial Immune Systems for Multispectral feature Extraction", Proc. Of SPIE Conf. on Algorithms for Multispectral and Hyperspectral Imagery IV, SPIE 3372, pp. 241-249.
16. Dasgupta D. and Forrest S. Novelty detection in time series data using ideas from immunology. ISCA (th Int. Conf. on Intelligent Systems. Reno, USA, 1996.
17. S. Forrest, S. Hofmeyr, A. Somayaji, and T. Longstaff. A sense of self for UNIX processes. In *Proceedings of the 1996 IEEE Symposium on Computer Security and Privacy*. IEEE Press, 1996.
18. D.Dasgupta, K.KrishnaKumar, D.Wong, M.Berry Immunity-Based Aircraft Fault Detection System. In the proceedings of American Institute of Aeronautics and Astronautics Chicago, USA September 20-24, 2004.
19. S. Forrest. A. S. Perelson, L. Allen, and R. Cherukuri. Self-Nonself Discrimination in a Computer. In Proceedings of IEEE Symposium on Research in Securitv and Privacy, pages202~212, Oakland, May 16-18 1994.
20. M. Ayara, J. Timmis, R. de Lemos. L. de Castro and R. Duncan, Negative Selection: How to Generate Detectors, 1st ICARIS, 2002
21. Fabio Gonzalez, Dipankar Dasgupta, An Immunogenetic Approach to Intrusion Detection, GECCO, 2002
22. Fabio Gonzalez, Dipankar Dasgupta. and Luis Fernando Nino, A Randomized Real-Value Negative Selection Algorithm, ICARIS-2003.
23. J. Kim amd P. J. Bentley, Toward an artificial immune system system for network intrusion detection: An investigation of dynamic Clonal selection, in Proceedings of the 2002 Congress on Evolutional Computation CEC2002. Honolulu. 2002
24. S. Hofmeyr and S. Forrest, Architecture for an artificial immune system, Evolutionary Compulation, vol. 8, no. 4, pp. 443-473. 2000

Получено 13.10.2005 г.