ISSN 1814-1196
Научный вестник НГТУ
том 62, № 1, 2016, с. 133–145

http://journals.nstu.ru/vestnik
Science Bulletin of the NSTU
Vol. 62, No. 1, 2016, pp. 133–145

СОВРЕМЕННЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

MODERN INFORMATION TECHNOLOGIES

УДК 004.43

Объектно-ориентированное проектирование нейронной сети для автоматизации определения архитектуры вычислительной системы в задачах обеспечения информационной безопасности*

Н.И. СЕЛЬВЕСЮК[1], А.С. ОСТРОВСКИЙ[2], Р.С. АРИСТОВ[3], А.А. ГЛАДКИХ[4]

[1] *105005, РФ, г. Москва, ул. 2-я Бауманская, 5, Московский государственный технический университет им. Н.Э. Баумана, доктор технических наук, доцент. E-mail: selvesyuk@yandex.ru*
[2] *105005, РФ, г. Москва, ул. 2-я Бауманская, 5, Московский государственный технический университет им. Н.Э. Баумана, кандидат технических наук. E-mail: aleksandr_ostrovsky@mail.ru*
[3] *107023, РФ, г. Москва, ул. Б. Семеновская, 45, ООО «ИНФОРИОН». E-mail: r.aristov@gmail.com*
[4] *105005, РФ, г. Москва, ул. 2-я Бауманская, 5, Московский государственный технический университет им. Н.Э. Баумана, кандидат технических наук. E-mail: gladkikhalexei@gmail.com*

Статья посвящена вопросам анализа бинарных данных, содержащихся во внутренней энергонезависимой памяти вычислительных систем. Показано, что при решении задач, предполагающих анализ бинарных данных, в условиях отсутствия технической документации на целевую вычислительную систему возникает необходимость определения архитектуры вычислительной системы. Отмечено, что отсутствие методов и средств автоматизации определения архитектуры вычислительной системы приводит к повышению требований к квалификации аналитика, увеличению временных затрат и снижению достоверности. Таким образом, очевидна актуальность решения задачи разработки средств автоматизации определения архитектуры вычислительной системы по имеющимся бинарным данным. Обоснована необходимость использования математического аппарата нейронных сетей. Исходя из анализа предметной области установлено, что в полном объеме реализовать процессы обучения и применения по назначению при решении поставленной задачи позволят рекуррентные нейронные сети. Обоснована необходимость проектирования программного изделия, реализующего рекуррентную нейронную сеть и предназначенного для автоматизации определения архитектуры вычислительной системы по имеющимся бинарным данным. Предложено использование объектно-ориентированного подхода при проектировании указанного программного изделия. Сформулированы основные задачи, решаемые изделием: создание рекуррентных нейронных сетей, настройка параметров оптимизации, интерпретация значений выходного слоя и вычисление

* *Статья получена 23 октября 2015 г.*

Работа выполнена при финансовой поддержке РФФИ (гранты № 14-08-00640а, 16-08-00311а).

функции потерь. Представлены диаграммы компонентов, классов и состояний в нотации языка объектно-ориентированного моделирования UML, а также приведены в текстовом виде назначения элементов диаграмм. Для демонстрации подхода выполнено объектно-ориентированное проектирование программного изделия, реализующего рекуррентную нейронную сеть и предназначенного для автоматизации определения архитектуры вычислительных систем. Изделие может быть использовано в ходе решения ряда задач, связанных с анализом бинарных данных, в интересах обеспечения информационной безопасности вычислительных систем.

Ключевые слова: информационная безопасность, объектно-ориентированное проектирование, вычислительные системы, нейронные сети, UML, анализ бинарных данных, дизассемблирование, нечеткое сравнение данных

DOI: 10.17212/1814-1196-2016-1-133-145

ВВЕДЕНИЕ

При решении ряда задач, связанных с аудитом информационной безопасности функционального и микропрограммного обеспечения вычислительных систем, возникает необходимость анализа бинарных данных, содержащихся в энергонезависимой памяти этих систем. Существующие методики анализа бинарных данных, направленные на поиск в этих данных заданных сигнатур, определение форматов файлов и разбор их структуры, дизассемблирование, сравнение и поиск схожих бинарных данных, предполагают использование сведений об архитектуре целевой вычислительной системы [1]. Однако при проведении аудита информационной безопасности в случае отсутствия технической документации появляется необходимость анализа бинарных данных в условиях неосведомленности об архитектуре вычислительной системы. В этом случае задача определения архитектуры решается аналитиком экспертным методом. Следовательно, отсутствие методов и средств автоматизации определения архитектуры вычислительной системы приводит к повышению требований к квалификации аналитика, увеличению временных затрат и снижению достоверности. Таким образом, задача разработки средств автоматизации определения архитектуры вычислительной системы по имеющимся бинарным данным является актуальной.

1. ПОСТАНОВКА ЗАДАЧИ И ВЫБОР МАТЕМАТИЧЕСКОГО АППАРАТА

Анализ предметной области показывает, что задача автоматизации определения архитектуры вычислительной системы по имеющимся бинарным данным может быть решена посредством сравнения этих данных с эталонными, однозначно идентифицирующими архитектуру вычислительной системы.

При решении аналогичных задач для сравнения данных используются метод сигнатурного анализа и математический аппарат нейронных сетей. Преимущество последнего перед сигнатурным анализом бинарных данных заключается в возможности нечеткого сравнения. Иначе говоря, сигнатурный анализ предполагает «жесткое» сравнение однозначно заданных сигнатур (последовательностей байт), а нейронные сети позволяют осуществлять поиск «похожих» данных.

При использовании сигнатурного анализа зачастую задать уникальную и неразрывную последовательность байт, полностью идентифицирующую необходимые данные, достаточно сложно. Если последовательность будет слишком короткой, то возникнет большое количество ложных срабатываний, если же слишком длинной, то в результате может быть получена ситуация, когда в заданной последовательности есть «незначащие» данные (те данные, которые не определяют искомую информацию и могут меняться). В таком случае необходимо привлекать аналитика для исследования форматов хранения данных и определения последовательности сигнатур. Следовательно, метод сигнатурного анализа является неприменимым для решения задачи автоматизации определения архитектуры вычислительной системы.

Применение математического аппарата нейронных сетей позволяет использовать унифицированный процесс определения формата и структуры информации на этапе обучения сети, так как для этого применяются одинаковые алгоритмы оптимизации при любом из форматов данных. На этапе обучения нейронная сеть самостоятельно выстраивает свои весовые коэффициенты так, чтобы соответствовать искомым данным при ее использовании. Причем на выходе нейронной сети будут получены вероятностные оценки соответствия текущих бинарных данных архитектурам вычислительных систем, на которые нейронная сеть была обучена. Следовательно, появляется возможность дополнительной обработки результатов работы нейронной сети с целью получения более детальной информации для анализа.

Таким образом, для решения задачи автоматизации определения архитектуры вычислительной системы предлагается использовать математический аппарат нейронных сетей.

2. ВЫБОР ТИПА НЕЙРОННОЙ СЕТИ И АНАЛИЗ ПРИНЦИПОВ ЕЕ РАБОТЫ

Наиболее простые в реализации нейронные сети прямого распространения были разработаны в результате исследования процессов, происходящих в биологических нейронах. Благодаря своей архитектуре, они обладают рядом преимуществ перед традиционными методами обработки информации: параллельность вычислений и устойчивость к повреждению части информации.

Основным элементом искусственной нейронной сети прямого распространения является искусственный нейрон [2]. Структура искусственного нейрона приведена на рис. 1.

При прямом распространении сигнала сначала вычисляется активационный потенциал, представляющий собой взвешенную сумму всех входов. Как правило, к нему также добавляется некоторое значение смещения, позволяющее нейрону иметь отличный от нуля активационный потенциал даже при равенстве нулю всех входных значений. После от активационного потенциала вычисляется значение активационной функции, например, гиперболического тангенса. Эта функция выбирается таким образом, чтобы отображать произвольную область входных значений на ограниченный интервал выходных, допустим (–1, 1). Так как в процессе оптимизации весов обычно используются градиентные методы, для эффективного вычисления градиентов методом обратного распространения ошибки необходимо, чтобы производная

функции активации могла быть выражена через ее значение (в случае с гиперболическим тангенсом $tanh'x = 1 - tanh^2 x$).

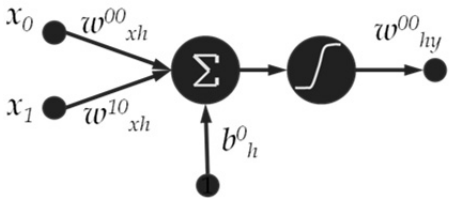

Рис. 1. Структура одиночного искусственного нейрона

Нейроны соединяются в многослойную структуру, обычно используя соединение «каждый с каждым», в результате чего получается искусственная нейронная сеть прямого распространения [2]. Структура искусственной нейронной сети прямого распространения показана на рис. 2.

При использовании нейронной сети для решения задач классификации входные данные представляются в некотором удобном для интерпретации сетью виде и подаются на входы сети. Выходные данные интерпретируются в соответствии с моделью обучения. Так, например, при распознавании изображений они могут быть преобразованы в вектор значений яркости пикселей, а обучение сети будет проводиться таким образом, что выходное значение на k-м выходе будет интерпретировано как вероятность соответствия изображения k -му классу.

Таким образом, задача обучения сводится к оптимизации функции ошибки (также называемой функцией потерь – Loss Function), показывающей, насколько выходные значения сети отличаются от ожидаемых на множестве весовых коэффициентов.

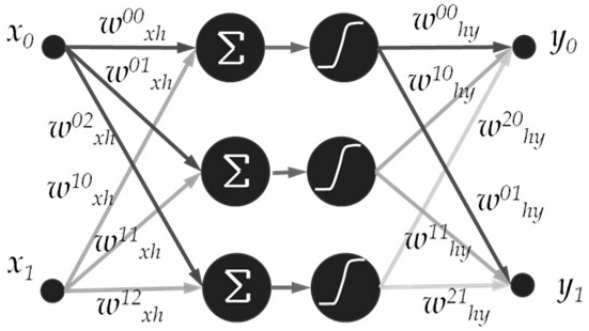

Рис. 2. Структура искусственной нейронной сети прямого распространения

Описанный подход обладает существенным недостатком при применении к задаче классификации последовательностей. В связи с тем, что выходные значения нейронных сетей прямого распространения полностью определяются набором входных значений, сеть не имеет возможности запомнить контекст, т. е. какие-либо свойства информации, представленной ей ранее. Однако эта информация может быть необходима для правильной классифи-

кации и определения архитектуры вычислительной системы. Одним из вариантов решения является предоставление сети требуемого контекста в виде движущегося по последовательности «окна», так что в каждый момент времени на вход сети будет подана как информация о текущем элементе последовательности, так и об N предыдущих. Данный подход не позволяет решить задачу эффективно, так как длина контекста ограничивается N значениями, где N различно для каждого наблюдения.

Для решения этой задачи применяются рекуррентные нейронные сети [3], которые строятся по архитектуре, сходной с архитектурой сетей прямого распространения, но имеют обратную связь в скрытом слое, позволяющую им запоминать контекст. Структура нейрона рекуррентной нейронной сети показана на рис. 3.

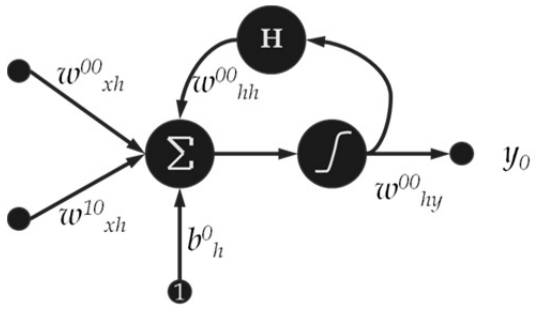

Рис. 3. Структура нейрона искусственной рекуррентной нейронной сети

Так как выходное значение на каждом шаге определяется не только взвешенной суммой входных значений, но и взвешенной суммой предыдущих выходов скрытого слоя, данная архитектура естественным образом может обрабатывать последовательные данные. Для этого достаточно подавать элементы последовательности по одному, каждый раз запоминая состояние скрытого слоя и используя его в вычислении следующего выходного значения.

Как и в случае с сетью прямого распространения, задача обучения рекуррентной нейронной сети является задачей оптимизации функции ошибки – поиска соответствующего набора весовых коэффициентов, при которых функция ошибки на всем множестве обучающих примеров оказывается минимальной [3]. На рис. 4 показана схема обучения такой рекуррентной нейронной сети.

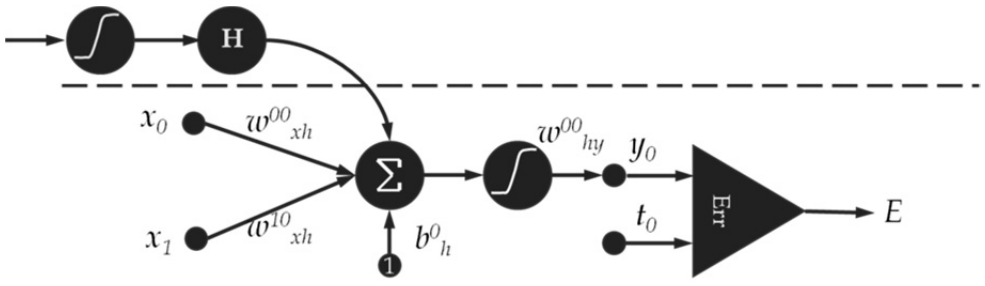

Рис. 4. Схема обучения рекуррентной нейронной сети

Для оптимизации функции ошибки, как правило, используются методы, основанные на градиентном спуске, так как значение градиента функции ошибки по каждому из весов показывает, насколько функция ошибки изменится при бесконечно малом изменении соответствующего веса. Однако следует отметить, что обратное распространение следует проводить не только от последних слоев к первым, но и от самых поздних элементов последовательности к самым ранним.

Таким образом, для решения задачи автоматизации определения архитектуры вычислительной системы возникает необходимость проектирования программного изделия, обеспечивающего создание рекуррентных нейронных сетей, настройку параметров оптимизации, интерпретацию значений выходного слоя и вычисление функции потерь.

3. ПРОЕКТИРОВАНИЕ РЕКУРРЕНТНОЙ НЕЙРОННОЙ СЕТИ

В настоящее время используется два основных подхода к проектированию программных изделий:

функционально-модульный, основанный на функциональной декомпозиции, при которой структура изделия описывается в терминах иерархии его функций и иерархии структур данных;

объектно-ориентированный, использующий объектную декомпозицию, при которой структура изделия описывается в терминах объектов и связей между ними, а его поведение – в терминах обмена сообщениями между объектами.

Достоинством второго подхода является то, что есть единая иерархия и нет необходимости отслеживать соответствие между двумя иерархиями функционально-модульного подхода [4].

Таким образом, для проектирования программного изделия использован объектно-ориентированный подход. Предлагаются следующие диаграммы проектируемого программного изделия в нотации языка UML, наиболее полно описывающие его структуру и поведение: диаграмма компонентов, диаграмма классов, диаграмма состояний.

На рис. 5 представлена диаграмма компонентов проектируемого программного изделия. Основными архитектурными компонентами изделия являются следующие блоки:

RNN Core – компонент-ядро изделия для работы с нейронными сетями;

Fast Function Lib – компонент, представляющий собой набор функций для увеличения скорости вычислений;

Optimizer – компонент для оптимизации весовых коэффициентов рекуррентной нейронной сети;

BLAS – внешний компонент, использующийся для быстрых вычислений.

Диаграмма классов проектируемого программного изделия представлена на рис. 6. Основными классами изделия являются BLAS, Matrix, NeuroWeight, RNN, NeuralNet, OptimizerBase, RMSProsOptmizer, AdagradOptimizer. Далее рассмотрим подробнее каждый из приведенных классов.

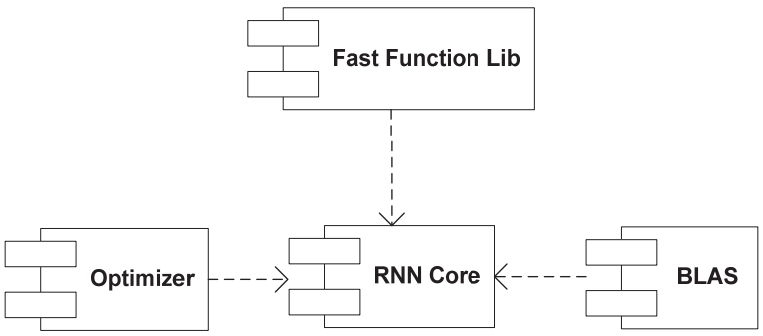

Рис. 5. Диаграмма компонентов проектируемого программного изделия

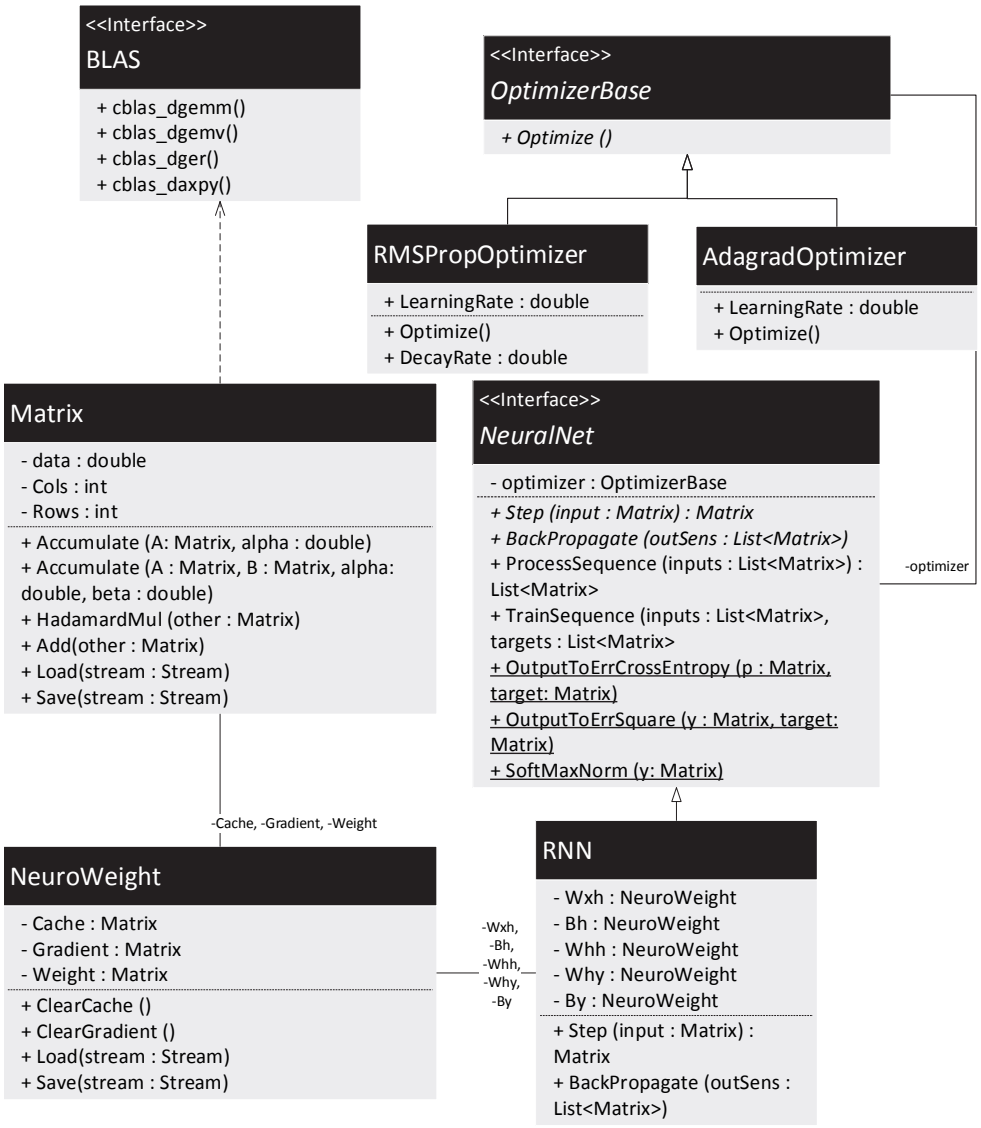

Рис. 6. Диаграмма классов проектируемого программного изделия

Так как в основе вычислений прямого и обратного проходов нейронной сети лежат матричные операции (и скорость их выполнения преимущественно определяет общую производительность системы), первым был описан класс, реализующий работу с матрицами через интерфейс BLAS (Basic Linear Algebra Subprograms), ставший стандартом де-факто для вычислений линейной алгебры. Благодаря использованию BLAS, представляется возможным использовать сторонние оптимизированные библиотеки для ускорения вычислений, такие как открытая реализация OpenBLAS и оптимизированная для многоядерных процессоров Intel MKL (Math Kernel Library).

Класс Matrix представляет собой матрицу чисел с плавающей запятой двойной точности и реализует операции умножения матрицы на число или на матрицу с накоплением, сложения матриц, поэлементного перемножения матриц, а также инициализацию матрицы случайными значениями с заданной дисперсией, их потоковую загрузку и сохранение. Математические операции сводятся к подготовке данных и вызовам функций из выбранной реализации BLAS.

Абстрактный класс NeuralNet предоставляет стандартный интерфейс нейронной сети: функции сохранения и загрузки из файла текущего состояния нейронной сети, функцию единичного шага прямого распространения Step (принимающую на вход матрицу входных значений и выдающую на выходе матрицу ответа сети), функцию обратного распространения ошибки во времени и статические функции интерпретации выходного значения (вычисление кросс-энтропийной функции потери, нормализации выходной матрицы методом SoftMax).

Класс NeuroWeight реализует обобщенную структуру весовых связей нейронной сети, имея в своем составе как сами значения весов, так и значения текущих градиентов функции ошибки относительно каждого из них и *кеш* изменений веса для более сложных алгоритмов оптимизации. Также данный класс позволяет сохранять и загружать данные о весах и градиентах из потока ввода-вывода. Класс RNN представляет собой реализацию рекуррентной нейронной сети, наследуя интерфейс NeuralNetwork. В своем составе он имеет набор весов, соответствующий архитектуре рекуррентной нейронной сети и реализацию функций интерфейса Step, Backropagate и Optimize, отвечающих за единичный шаг в прямом направлении, обратное распространение ошибки во времени по последовательности и оптимизацию весов в соответствии с накопленными значениями градиента, соответственно.

Таким образом, благодаря определенным в классе рекуррентной нейронной сети методам Step и Backpropagate становится возможным использовать более высокоуровневые методы, определенные в интерфейсе, – ProcessSequence и TrainSequence, предназначенные для обработки последовательностей. Эти методы принимают на вход список матриц, соответствующих входным значениям нейронов на каждом шаге (в случае TrainSequence – вместе со списком ожидаемых выходных значений). ProcessSequence возвращает список выходных значений сети, в то время как TrainSequence по списку ожидаемых значений накапливает градиент функции ошибки относительно каждого из весов методом обратного распространения ошибки во времени, подготавливая данные для оптимизатора. Оптимизатор может быть вызван функцией

Optimize, принимающей на вход вещественные параметры rate и decay, регулирующие скорость оптимизации и позволяющие избежать локальных минимумов функции ошибки.

Класс OptimizerBase реализуют единый интерфейс для различных алгоритмов оптимизации. В проектируемом программном изделии предполагается использовать два основных алгоритма оптимизации с помощью классов RMSProsOptimizer и AdagradOptimizer.

Диаграмма состояний показана на рис. 7. Как видно из диаграммы, возможно два основных режима работы – это обучение нейронной сети на заданных бинарных данных, соответствующих определенной архитектуре вычислительной системы, и непосредственно режим поиска бинарных данных с помощью нейронной сети. Причем первоначальным режимом работы нейронной сети является ее обучение, после которого возможно применение на заданных данных. Рассмотрим каждый из приведенных режимов работы.

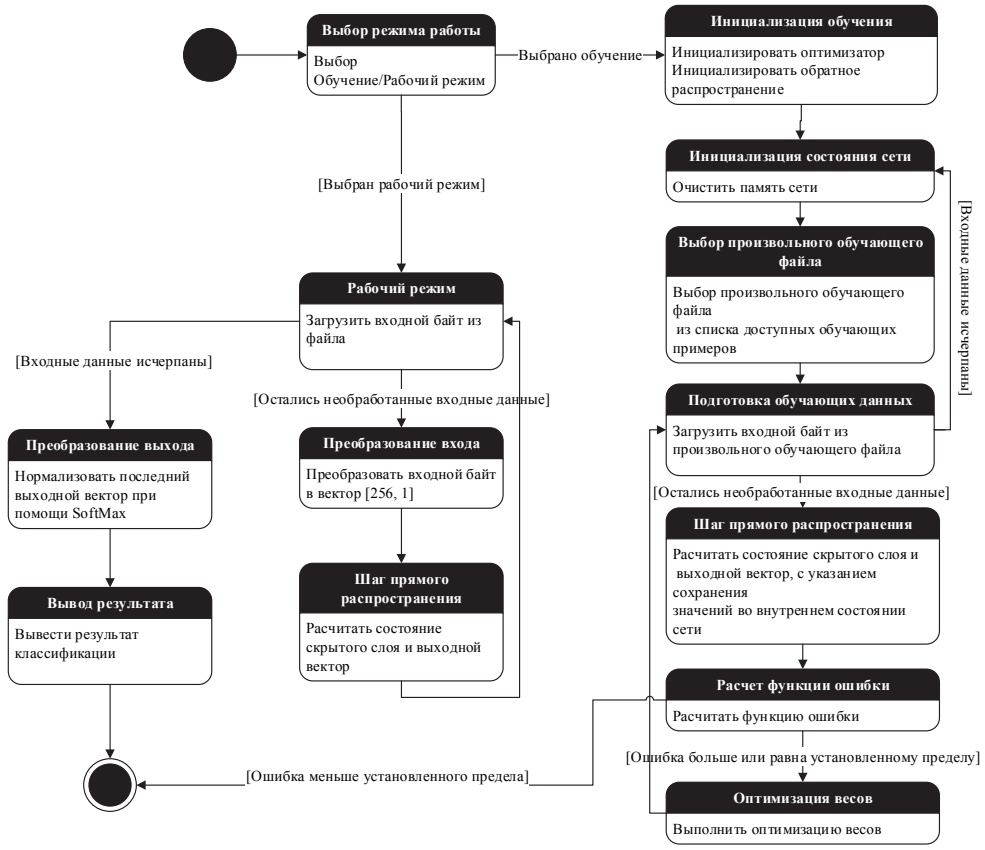

Рис. 7. Диаграмма состояний проектируемого программного изделия

В режиме обучения нейронная сеть проходит несколько состояний. «Инициализация обучения» предполагает инициализацию оптимизатора, который будет использоваться для обучения рекуррентной нейронной сети. В состоянии «Инициализация сети» создаются начальные значения весовых коэффициентов сети. Далее идет состояние подготовки обучающих данных

из заранее созданных пользователем. Следующие состояния – это прямое распространение и вычисление ошибки, согласно которой корректируются весовые коэффициенты сети. После того как рекуррентная нейронная сеть достигнет достаточного уровня обучения, ее можно использовать для определения архитектуры вычислительной системы.

Другим режимом работы сети является непосредственно обработка поданных на ее вход массивов данных. При подаче на нейронную сеть бинарных данных последовательным образом (к примеру, по байтам) обученная нейронная сеть будет выдавать вероятность того, встречаются ли данные, на которых проводилось обучение, для обработанной информации. Таким образом, выставив некоторый порог по вероятности можно отследить те области входных данных, которые могут соответствовать искомым.

ЗАКЛЮЧЕНИЕ

В статье предложен подход и представлены результаты объектно-ориентированного проектирования программного изделия, реализующего рекуррентную нейронную сеть и предназначенного для автоматизации определения архитектуры вычислительных систем. Представлены диаграммы компонентов, классов и состояний в нотации языка объектно-ориентированного моделирования UML, а также приведены в текстовом виде назначения элементов диаграмм.

Программное изделие может быть использовано в ходе решения ряда задач, связанных с анализом бинарных данных, в интересах обеспечения информационной безопасности вычислительных систем.

СПИСОК ЛИТЕРАТУРЫ

1. *Гетьман А.И., Падарян В.А*. Восстановление формата данных путем анализа бинарного кода: состояние и перспективы // Проблемы информационной безопасности. Компьютерные системы. – 2014. – № 3. – С. 123–130.

2. *Падарян В.А., Каушан В.В., Федотов А.Н.* Автоматизированный метод построения эксплойтов для уязвимости переполнения буфера на стеке // Труды Института системного программирования РАН. – 2014. – Т. 26, вып. 3. – С. 127–144.

3. Метод выявления некоторых типов ошибок работы с памятью в бинарном коде программ / В.В. Каушан, А.Ю. Мамонтов, В.А. Падарян, А.Н. Федотов // Труды Института системного программирования РАН. – 2015. – Т. 27, вып. 2. – С. 105–126.

4. *Тихонов А.Ю., Аветисян А.И.* Комбинированный (статический и динамический) анализ бинарного кода // Труды Института системного программирования РАН. – 2012. – Т. 22. – С. 131–152.

5. Применение программных эмуляторов в задачах анализа бинарного кода / П.М. Довгалюк, В.А. Макаров, В.А. Падарян, М.С. Романеев, Н.И. Фурсова // Труды Института системного программирования РАН. – 2014. – Т. 26, вып. 1. – С. 277–296.

6. *Круглов В.В., Дли М.И., Горбунов Р.Ю.* Нечеткая логика и искусственные нейронные сети. – М.: Физматлит, 2001. – 224 с.

7. *Graves A.* Multi-dimensional recurrent neural networks // Lecture Notes in Computer Science. – 2007. – Vol. 4668. – P. 549–558 p.

8. *Бендерская Е.Н., Никитин К.В.* Рекуррентная нейронная сеть как динамическая система и подходы к ее обучению // Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. Информатика. Телекоммуникации. Управление. – 2013. – № 176. – С. 29–40.

9. *Меркушева А.В., Малыхина Г.Ф.* Нейронная сеть с множественной рекуррентной структурой // Научное приборостроение. – 2012. – Т. 22, № 3. – С. 107–113.

10. *Будыльский Д.В.* Применение рекуррентных нейронных сетей в задачах обработки текстов на естественном языке // Вопросы науки. – 2015. – Т. 6. – С. 8–12.

11. *Пучков Е.В., Лила В.Б.* Методология обучения рекуррентной искусственной нейронной сети с динамической стековой памятью // Программные продукты и системы. – 2014. – № 4 (108). – С. 132–135.

12. *Рамбо Дж., Блаха М.* UML 2.0. Объектно-ориентированное моделирование и разработка: пер. с англ. – 2-е изд. – СПб.: Питер, 2006. – 544 с.

13. *Буч Г., Якобсон А., Рамбо Дж.* UML: пер. с англ. / под общ. ред. С. Орлова. – 2-е изд.– СПб.: Питер, 2006. – 736 с. – (Классика CS).

14. *Фаулер М.* UML. Основы: краткое руководство по стандартному языку объектного моделирования: пер. с англ. – 3-е изд. – СПб.: Символ, 2009. – 192 с.

15. *Вендров А.М.* Проектирование программного обеспечения экономических информационных систем. – М.: Финансы и статистика, 2006. – 544 с.

Сельвесюк Николай Иванович, доктор технических наук, профессор кафедры «Информационная безопасность» Московского государственного технического университета им. Н.Э. Баумана. Основные направления научных исследований: устойчивость систем, информационная безопасность, дистанционное зондирование Земли. Имеет более 150 публикаций. E-mail: selvesyuk@yandex.ru

Островский Александр Сергеевич, кандидат технических наук, доцент кафедры «Информационная безопасность» Московского государственного технического университета им. Н.Э. Баумана. Основные направления научных исследований: информационная безопасность, дистанционное зондирование Земли. Имеет более 100 публикаций. E-mail: aleksandr_ostrovsky@mail.ru

Гладких Алексей Алексеевич, кандидат технических наук, доцент кафедры «Проектирование и технология производства электронной аппаратуры» Московского государственного технического университета им. Н.Э. Баумана. Основные направления научных исследований: информационная безопасность, электронно-вычислительная техника. Имеет 18 публикаций. E-mail: gladkikhalexei@gmail.com

Аристов Роман Сергеевич, ведущий специалист отдела исследований и разработок ООО «ИНФОРИОН». Основные направления научных исследований: информационная безопасность, электронно-вычислительная техника, нейронные сети. r.aristov@gmail.com

Object-oriented design of a neural network to automate the process of computer architecture determination in the information security problems[*]

N.I. SELVESYUK[1], *A.S. OSTROVSKIY*[2], *R.S. ARISTOV*[3], *A.A. GLADKIKH*[4]

[1] *Bauman Moscow State Technical University, 5 2-nd Bauman Street, Moscow, 105005, Russian Federation, D. Sc. (Eng.), associate professor. E-mail: selvesyuk@yandex.ru*

[2] *Bauman Moscow State Technical University, 5 2-nd Bauman Street, Moscow, 105005, Russian Federation, PhD (Eng.). E-mail: aleksandr_ostrovsky@mail.ru*

[3] *INFORION, LLC, 45 B. Semenovskaya st., Moscow, 107023, Russian Federation. E-mail: r.aristov@gmail.com*

[4] *Bauman Moscow State Technical University, 5 2-nd Bauman Street, Moscow, 105005, Russian Federation, PhD (Eng.). E-mail: gladkikhalexei@gmail.com*

The article is devoted to the analysis of binary data contained in the internal non-volatile computer system memory. It is proved that while solving problems involving the analysis of binary data in the absence of technical documentation on the target computer system it is necessary to determine the architecture of a computer system. It is noted that the lack of methods and

[*] *Received 23 October 2015.*
The work was supported by RFBR (grant number 14-08-00640a, 16-08-00311a).

tools for automating the definition of the computer system architecture leads to increased requirements to the skills of the analyst as well as a temporary increase in costs and decrease in reliability. Thus, we have proved the urgency of solving problems of developing automation means for determining the computer system architecture based on the binary data available. The necessity of using the mathematical apparatus of neural networks has been validated. Based on the domain analysis it is found that fully realize the intended learning and application processes in the solution of the problem will allow recurrent neural networks. The necessity of developing a software product that implements a recurrent neural network and is designed to automate the definition of the computer system architecture based on to the binary data available. It is proposed to use an object-oriented approach to the design of this software product. The main tasks to be solved by the product include creating recurrent neural networks, setting optimization parameters as well as the interpretation of the output layer values and the calculation of the loss function. The component, class and status diagrams in the object-oriented modeling language of the UML notation are presented and allocations of chart elements are listed in a tabular format. To demonstrate the approach an object-oriented design software product that implements the recurrent neural network and is designed to automate the definition of the computer architecture is proposed. The product can be used in of solving a number of problems associated with the analysis of binary data in order to ensure the information security of computer systems.

Keywords: Information security; object-oriented design; computer systems; neural networks; UML, analysis of binary data; disassembling; fuzzy comparison data

DOI: 10.17212/1814-1196-2016-1-133-145

REFERENCES

1. Getman A.I., Padaryan V.A. Vosstanovlenie formata dannykh putem analiza binarnogo koda: sostoyanie i perspektivy [Data format recovery through binary code analysis: state and perspectives]. *Problemy informatsionnoi bezopasnosti. Komp'yuternye sistemy – Information Security Problems. Computer Systems*, 2014, no. 3, pp. 123–130.

2. Padaryan V.A., Kaushan V.V., Fedotov A.N. Avtomatizirovannyi metod postroeniya eksploitov dlya uyazvimosti perepolneniya bufera na steke [Automated exploit generation method for stack buffer overflow vulnerabilities], *Trudy Instituta sistemnogo programmirovaniya RAN – Programming and Computer Software*, 2014, vol. 26, iss. 3, pp. 127–144. (In Russian)

3. Kaushan V.V., Mamontov A.Ju, Padaryan V.A., Fedotov A.N. Metod vyyavleniya nekotorykh tipov oshibok raboty s pamyat'yu v binarnom kode programm [Memory violation detection method in binary code]. *Trudy Instituta sistemnogo programmirovaniya RAN – Programming and Computer Software*, 2015, vol. 27, iss. 2, pp. 105–126. (In Russian)

4. Tikhonov A.Yu., Avetisyan A.I. Kombinirovannyi (staticheskii i dinamicheskii) analiz binarnogo koda [Combined (static and dynamic) analysis of binary code]. *Trudy Instituta sistemnogo programmirovaniya RAN – Programming and Computer Software*, 2012, vol. 22, pp. 131–152. (In Russian)

5. Dovgalyuk P.M., Makarov V.A., Padaryan V.A., Romaneev M.S., Fursova N.I. Primenenie programmnykh emulyatorov v zadachakh analiza binarnogo koda [Application of software emulators for the binary code analysis]. *Trudy Instituta sistemnogo programmirovaniya RAN – Programming and Computer Software*, 2014, vol. 26, iss. 1, pp. 277–296. (In Russian)

6. Kruglov V.V., Dli M.I., Gorbunov R.Yu. *Nechetkaya logika i iskusstvennye neironnye seti* [Fuzzy logic and artificial neural network]. Moscow, Fizmatlit Publ., 2001. 224 p.

7. Graves A. Multi-dimensional recurrent neural networks. *Lecture Notes in Computer Science*, 2007, vol. 4668, pp. 549–558.

8. Benderskaya E.N., Nikitin K.V. Rekurrentnaya neironnaya set' kak dinamicheskaya sistema i podkhody k ee obucheniyu [Recurrent neural network as dynamical system and approaches to its training]. *Nauchno-tekhnicheskie vedomosti Sankt-Peterburgskogo gosudarstvennogo politekhnicheskogo universiteta. Informatika. Telekommunikatsii. Upravlenie – St. Petersburg State Polytechnic University Journal. Computer science. Telecommunications and Control Systems*, 2013, vol. 176, pp. 29–40.

9. Merkusheva A.V., Malykhina G.F. Neironnaya set' s mnozhestvennoi rekurrentnoi strukturoi [Neural network with manifold recurrent structure]. *Nauchnoe priborostroenie – Scientific Instrumentation*, 2012, vol. 22, no. 3, pp. 107–113.

10. Budyl'skii D.V. Primenenie rekurrentnykh neironnykh setei v zadachakh obrabotki tekstov na estestvennom yazyke [The use of recurrent neural networks in problems of word processing in natural language]. *Voprosy nauki – Science Issues*, 2015, vol. 6, pp. 8–12.

11. Puchkov E.V., Lila V.B. Metodologiya obucheniya rekurrentnoi iskusstvennoi neironnoi seti s dinamicheskoi stekovoi pamyat'yu [Methodology of training recurrent artificial neural network with dynamic stack memory]. *Programmnye produkty i sistemy – Software and Systems*, 2014, no. 4 (108), pp. 132–135.

12. Rumbaugh J., Blaha M. *Object oriented modeling and design with UML*. 2nd ed. Upper Saddle River, NJ, Pearson Education, 2005. 496 p. (Russ. ed.: Rambo Dzh., Blakha M. *UML 2.0. Ob"ektno-orientirovannoe modelirovanie i razrabotka*. 2nd ed. Translation from English. St. Petersburg, Piter Publ., 2007. 544 p.).

13. Rumbaugh J. Jacobson I., Booch G. *The unified modeling language reference manual*. 2nd ed. Boston, Addison-Wesley, 2005. 721 p. (Russ. ed.: Buch G., Yakobson A., Rambo Dzh. *UML*. 2nd ed. Translation from English. St. Petersburg, Piter Publ., 2006. 736 p.).

14. Fowler M. *UML Distilled*: *a brief guide to the standard object modeling language*. 3rd ed. Boston, Addison-Wesley, 2004. 175 p. (Russ. ed.: Fauler M. *UML. Osnovy: kratkoe rukovodstvo po standartnomu yazyku ob"ektnogo modelirovaniya*. 3rd ed. Translation from English. St. Petersburg, Symvol Publ., 2009. 192 p.).

15. Vendrov A.M. *Proektirovanie programmnogo obespecheniya ekonomicheskikh informatsionnykh sistem* [Software design of economic information systems]. Moscow, Finansy i statistika Publ., 2006. 544 p.