Дяченко Алла Вячеславовна, кафедра литейного производства, Национальный технический университет «Харьковский политехнический институт», ул. Багалия, 21, г. Харьков, Украина, 61002

Михайлова Анастасия Александровна, кафедра литейного производства, Национальный технический университет «Харьковский политехнический институт», ул. Багалия, 21, г. Харьков, Украина, 61002

УДК 004.85, 004.89
DOI: 10.15587/2313-8416.2016.69588

КРОСПЛАТФОРМЕННАЯ C++ БИБЛИОТЕКА ДЛЯ ОБУЧЕНИЯ МНОГОСЛОЙНОГО ПЕРЦЕПТРОНА

© Д. Т. Ибадов, И. В. Афанасьева

Разработана кроссплатформенная C++ библиотека, предоставляющая классы для создания многослойного перцептрона и его обучения с учителем методом обратного распространения ошибки, способного классифицировать входящие образцы после предварительного обучения. Протестированы результаты обучения многослойного перцептрона и его способность к классификации на пробной выборке данных
Ключевые слова: *искусственные нейронные сети, обучение, классификация, обратное распространение ошибки, многослойный перцептрон*

Cross-platform C++ library is developed. It provides classes enabling to create multilayer perceptron and its training by supervised learning method (backpropagation). Resulting artificial network is able to classify incoming data after previous training. Multilayer perceptron training results and its ability to classify test dataset was tested
Keywords: *artificial neural networks, training, classification, backpropagation, multilayer perceptron*

1. Введение

Анализ данных является важной вспомогательной областью для множества других. Сфера применения различных методов анализа данных включает в себя и академические, и прикладные цели. Интеллектуальный анализ данных (data mining) проводится для выявления неявных закономерностей в массивах данных, а следовательно, и реальных явлений, которые они описывают. Другое распространенное применение включает в себя распознание обстановки с целью принятия решений, причем непосредственно принимать решение может как пользователь, который получает информацию о результатах анализа с помощью устройств вывода (например, врач, пользующийся медицинской аппаратурой), так и автоматизированная система (например, автопилот).

С развитием вычислительной техники стало возможным применять машинное обучение в целях анализа данных во всё большем количестве систем и прикладных областей.

2. Анализ литературных данных и постановка проблемы

Искусственные нейронные сети (ИНС) – математические модели, конструкция которых была вдохновлена биологическими нейросетями, присущими центральной нервной системе (ЦНС) животного мира. В общих чертах, ИНС состоят из слоев нейронов, последовательно соединенных весами с различными коэффициентами. В сетях с прямой передачей сигнала (feed-forward) импульс (нормализованные входящие данные) подается на входной слой, затем переходит на следующий слой с умножением на коэффициенты соответствующих весов и так далее. Общий вход j-того нейрона net_j равен сумме импульсов всех связанных с ним нейронов предыдущего слоя (с d нейронов) с соответвующими коэффициентами и импульса с нейрона смещения w_{j0}:

$$net_j = \sum_{i=1}^{d} x_i w_{ji} + w_{j0}. \quad (1)$$

Каждый отдельный нейронный слой имеет свою активационную функцию f – операцию, которая вычисляется над суммарным входом на нейрон с предыдущего слоя перед тем, как сигнал z_j будет подан на выход:

$$z_j = f(net_j). \quad (2)$$

Построение моделей ИНС используется для решения широго ряда задач из различных областей, включающего в себя, например, компьютерное зрение, распознавание речи и прочее. Одно из регулярных применений ИНС – распознавание образов после предварительного обучения с учителем (классификация), когда ИНС присваивает класс входящему образцу на основании результатов обработки его параметров, которые зачастую представлены в нормализованном виде. Предварительное обучение производится на наборе образцов, классы которых уже известны. Сначала ИНС пытается самостоятельно определить класс объекта, затем результат её работы

сравнивается с известным классом ошибки, веса сети калибруются в случае необходимости.

Распространенной топологией ИНС является многослойный перцептрон – сеть прямой связи, которая состоит из входящего слоя, исходящего и одного или нескольких скрытых слоев между ними. Такие сети обладают высокой связностью, т. е. все нейроны соседних слоев связаны друг с другом (рис. 1).

В качестве активационной функции скрытого слоя используют нелинейные функции, поскольку не имеет смысла одновременно использовать более одного слоя с линейными функциями активации [1]. Активационные функции на исходящем слое обычно тоже нелинейные, но здесь они, в том числе, зависят от области значений, в которой должен находиться результат работы ИНС. Следовательно, активационные функции на скрытом и выходном слоях могут быть разными. Более того, иногда активационные функции нейронов одного слоя могут отличаться.

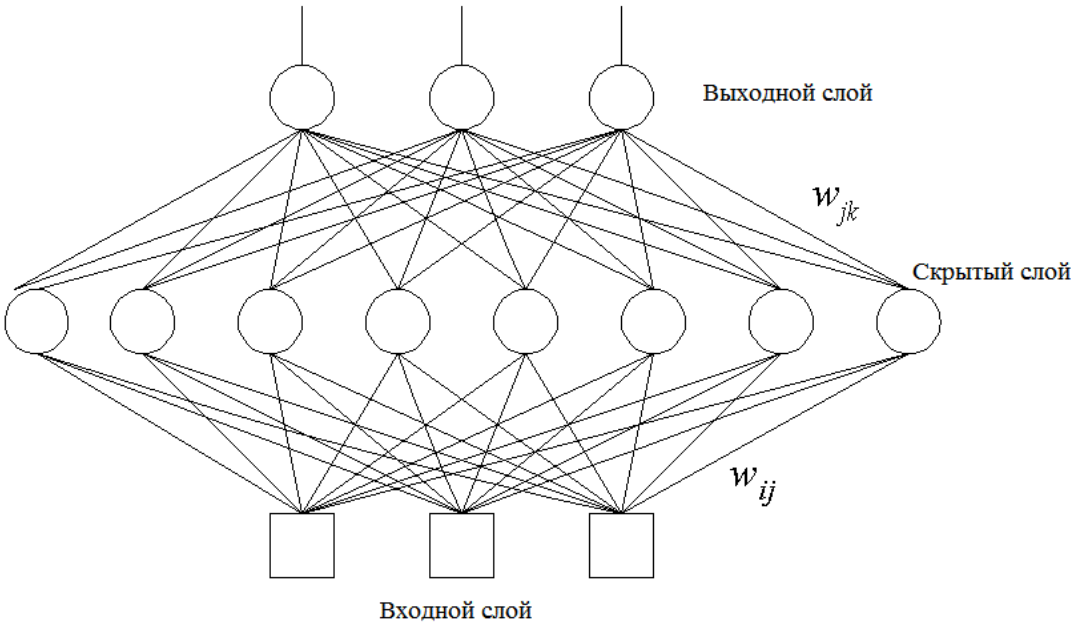

Рис. 1. Схема многослойного перцептрона, где w_{ij} – вес от i-го нейрона к j-ому

Другим преимуществом многослойных перцептронов является связанный с ними алгоритм обучения – обучение методом обратного распространения ошибки (backpropagation) [2]. Его суть состоит в вычислении ошибки сети во время обучения, а затем последовательной калибровке весов, соединяющих слои от конца к началу, в сторону уменьшения ошибки. Алгоритм относительно прост и нагляден в реализации, которая не требует значительных модификаций для применения многопоточных вычислений, что очень важно в виду направления развития современного аппаратного обеспечения.

Таким образом, хотя многослойные перцептроны имеют относительно простую топологию, набор задач, для решения которых они могут применяться, очень разнообразен, и включает в себя линейно-неразделимые задачи.

Классификация данных может быть не последним этапом в их обработке. Более того, обработанная информация может применяться для последующих вычислений, предоставления данных пользователю или принятия решений автоматизированными системами. Такие предпосылки показывают прикладную пользу в специализированном решении в виде встраиваемой в клиентский код библиотеки, с помощью которой можно создать многослойный перцептрон и использовать его для классификации данных. Все это обуславливает необходимость проведения работ в этом направлении, т. е. разработки соответствующей библиотеки и проверки ее работоспособности.

Библиотека должна быть кроссплатформенной, чтобы обеспечить возможность работы на максимальном количестве систем. Необходимо протестировать ее алгоритм обучения обратным распространением ошибки и способность к классификации, исследовать результаты, чтобы убедиться в возможности практической применимости.

3. Цели и задачи разработки

Цель работы – разработка программной библиотеки, предоставляющей классы для построения многослойного пецептрона с одним скрытым слоем, поскольку одного обычно хватает для обучения нейронной сети решения большинства проблем [3]. ИНС, созданная с помощью библиотеки, должна быть обучаема алгоритмом обратного распространения ошибки. Кроме того, библиотека должна предоставлять клиентскому коду возможность сохранять и загружать веса обученной сети, чтобы не проходить этап обучения каждый раз при её использовании.

Для достижения этой цели были решены следующие задачи:

– разработан объектный дизайн программной библиотеки;

– программно реализованы алгоритмы ИНС, в том числе алгоритм обратного распространения ошибки;

– проведено тестирование ИНС, созданной при помощи разработанной библиотеки, на двух выборках данных, для которых имеются примеры классификации с помощью других алгоритмов и технологий.

4. Разработка и методы тестирования

В качестве языка программирования был выбран C++. Это высокоуровневый язык, который позволяет использовать ООП-средства и средства функционального программирования. При этом язык обратно совместим с C и транслируется напрямую в байт-код, что обеспечивает относительно высокую скорость выполнения кода в виду отсутствия дополнительных промежуточных систем между выполняемым кодом и операционной системой.

Кроме того, обратная совместимость C++ с C, который является системным языком программирования, позволяет гарантировать возможность написания для C++ библиотеки специальных врапперов (“оберточных” программных модулей) для интеграции в программы на большинстве других языков программирования. Также C++ код обеспечивает кроссплатформенность библиотеки, т. е. ее потенциальную компилируемость под все распространенные архитектуры и операционные системы.

Библиотека позволяет создавать в клиентском коде многослойные перцептроны с одним скрытым слоем и произвольным числом нейронов на каждом слое. Активационная функция задается отдельно для скрытого и выходного слоев из трёх вариантов: линейная, сигмоидальная и гиперболическая тангенциальная.

Линейная активационная функция просто передает на выход сумму входа, ее можно использовать для тех случаев, когда результат классификации определен на непрерывной области значений:

$$f(net) = net\,. \qquad (3)$$

Сигмоидальная функция определена на промежутке (0; 1):

$$f(net) = \frac{1}{1+e^{-x}}\,. \qquad (4)$$

Ее удобно использовать для выявления, принадлежит ли входящий образец к какому-то классу. Аналогично может использоваться гиперболическая тангенциальная:

$$f(net) = \frac{2}{1+e^{-2x}} - 1\,. \qquad (5)$$

Она представляет собой масштабированную версию сигмоидальной функции, определена на промежутке (–1; 1).

Работоспособность разработанной библиотеки проверялась на двух выборках.

Первая выборка содержит статистику о показателях различных экземпляров ирисов и их типах. В ней имеется четыре параметра и зависимый от них классификатор, который может принимать три значения: Iris Setosa, Iris Versicolour и Iris Viginica.

Параметры по каждому образцу включают:

– длина листа чашечки цветка в см (s_len);

– ширина листка чашечки цветка в см (s_width);

– длина лепестка в см (p_len);

– ширина лепестка в см (p_width).

Выборка содержит 150 образцов. Все образцы класса Iris Setosa линейно отделимы от образцов двух других классов, но те, в свою очередь, линейно неотделимы друг от друга [4]. Для экспериментов образцы были перемешаны случайным образом, первые 100 были использованы для обучения сети, последние 50 – для её тестирования.

Для работы на этой выборке с помощью разработанной библиотеки был создан многослойный перцептрон с четырьмя нейронами на входном слое и тремя нейронами на выходном слое, каждый из которых обозначал принадлежность к одному из классов. Количество нейронов на скрытом слое варьировалось от эксперимента к эксперименту. Для активационной функции на скрытом и выходном слоях была выбрана сигмоидальная функция, чтобы перцептрон был способен решить линейно неразделимую задачу.

Вторая выборка содержит статистику диабета среди женщин североамериканского племени Пима. В выборке представлены данные о женщинах возрастом от 21 года и старше. Сама выборка содержит 8 параметров и зависимый от них бинарный классификатор, 1 свидетельствует о наличии диабета, 0 – об отсутствии. Классы линейно неотделимы друг от друга [5]. Таким образом, задача определения, больна ли женщина диабетом или нет, на данной выборке сводится к задаче классификации.

Параметры по каждому образцу включают:

– количество беременностей (npreg);

– концентрация глюкозы в плзаме (glu);

– диастолическое кровяное давление (bp);

– толщина кожной складки на трицепсе (skin);

– двухчасовая концентрация инсулина в сыворотке (ins);

– индекс массы тела (bpi);

– наследственная функция диабета (ped);

– возраст (age).

Для работы на такой выборке был создан многослойный перцептрон с восемью нейронами на входном слое и одним нейроном на выходном слое. Аналогично конфигурации сети для предыдущей выборки, количество нейронов на скрытом слое варьировалось от эксперимента к эксперименту. Также сигмоидальная функция была использована в качестве активационной на скрытом и выходном слоях.

Выборка состоит из 768 образцов, которые для экспериментов были перемешаны случайным образом. После этого 576 первых образцов были использованы для обучения сети, оставшиеся 192 образца – для тестирования.

Перед началом обучения входящие данные обеих выборок были нормализованы. Для этого каждый параметр каждого образца был высчитан по формуле:

$$X' = (X - \bar{X}) / \sigma_X\,, \qquad (6)$$

где X – изначальное значение параметра; X' – итоговое значение; $\overline{X}$ – среднее значение параметра на всей выборке данных; σ_x – среднеквадратичное отклонение. Такая нормализация позволила равномерно распределить значения параметров вокруг 0 и нивелировать разницу в их численных размерах. Так, во второй выборке npreg на порядок отличается от bp, и без нормализации их вклад в обучение сети был бы соответствующим, хотя они должны быть равновесными.

После нормализации с помощью разработанной библиотеки конструировались сети с указанными выше параметрами и переменным числом нейронов на скрытом слое.

Так как задачей тестирования являлось достижение наиболее высокой точности на тестовом наборе с минимальным числом нейронов на скрытом слое (чтобы минимизировать вычислительную сложность), то эксперименты начинались с сетями с 1 нейроном на скрытом слое, затем число нейронов увеличивалось.

Перед обучением все веса сети инициализировались случайными значениями в интервале (–0,25; 0,25). Этот этап необходим, так как текущий вес сети служит одним из коэффициентов во время калибровки сети с помощью алгоритма обратного распространения ошибки. Удачность инициализации весов сильно влияет на время обучения сети.

Обучение проводилось в режиме on-line, то есть изменения применялись к весам сети после каждого тренировочного образца. Также вычислялась средняя квадратичная ошибка (J) для каждой эпохи (полного прохода по всем тренировочным образцам) по следующей формуле:

$$J = \frac{1}{n} * \sum_{i=1}^{n} (t_i - z_i)^2 , \qquad (7)$$

где n – количество образцов в эпохе; t – ожидаемый выход сети; z – наблюдаемый выход сети.

После каждой эпохи проводилось тестирование сети на тестовом наборе. Если при этом достигалась точность выше, чем все предыдущие результаты в рамках данной попытки, то сконфигурированные веса сети сохранялись.

Так как начальные веса определяются случайно, а успешность обучения сети зависит от них, то обучение с каждой конфигурацией повторялось 10 раз, затем выбирался лучший результат.

Эксперименты сети на первой выборке проходили с коэффициентом скорости обучения 0.1, который был принят оптимальным после серии предварительных опытов, т. к. не останавливал обучение. Для этой выборки критерием остановки было достижение 100 % точности на тестовом наборе или 1000 эпохи (полного прохода по всем тренировочным образцам). На этой выборке эксперименты закончились на конфигурации сети с 5 нейронами на скрытом слое.

Эксперименты на второй выборке закончились с 20 нейронами на скрытом слое. Коэффициент скорости обучения был установлен в 0.06 – эта выборка более “сложная” для классификации, так как содержит больше образцов и параметров. Критерием остановки обучения было достижение J значения в 0.14, поскольку, как показала серия предварительных опытов при меньшем J сеть была чрезмерно специализирована, т. е. давала хороший результат на тренировочном наборе, но ухудшались её показатели на тестовом наборе.

5. Результаты исследований

В результате проведенных экспериментов был получен хороший результат.

Ниже приведены результаты обучения сети с различным количеством нейронов на скрытом слое в табл. 1.

Таблица 1

Результаты экспериментов с различной конфигурацией сети на первой выборке

Кол-во нейронов на скрытом слое	Кол-во эпох до достижения лучшего результата	Лучшая точность на тренировочном наборе
2	40	91 %
3	49	91 %
4	52	92 %
5	65	92 %

Точность на тестовом наборе для всех четырех приведенных в табл. 1 конфигураций была достигнута 100 %. Конфигурация сети для 1 нейрона на скрытом слое в табл. 1 не приведена, так как для нее была достигнута максимальная точность в 36 % на тестовом наборе. Из табл. 1 видна тенденция улучшения точности на тренировочном наборе по мере увеличения количества нейронов на скрытом слое. Возможно, можно было бы достичь точности, близкой к 100 % и на тренировочном наборе при большем количестве нейронов на скрытом слое, но т. к. главными показателями были выбраны точность на тестовом наборе и минимальное при этом число нейронов на скрытом слое, эксперименты дальше не продолжались.

Рис 2. демонстрирует изменение средней квадратической ошибки на протяжении обучения.

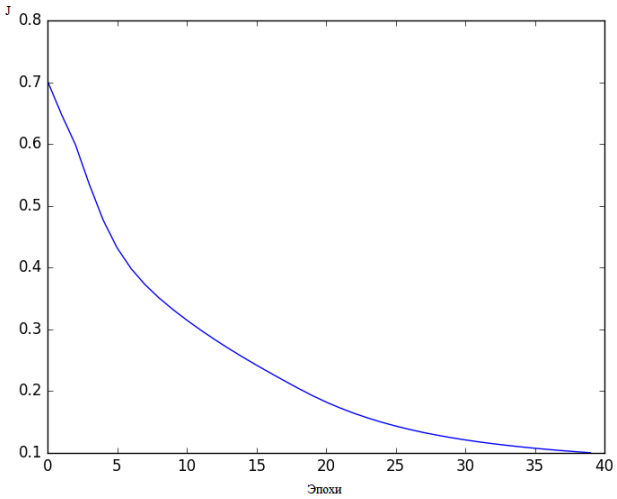

Рис. 2. Изменение средней квадратичной ошибки сети J на протяжении обучения на первой выборке

Из рис. 2 видно, что ошибка сети плавно понижалась на протяжении обучения. Этот график можно сопоставить с графиком увеличения точности сети на тестовом наборе (рис. 3):

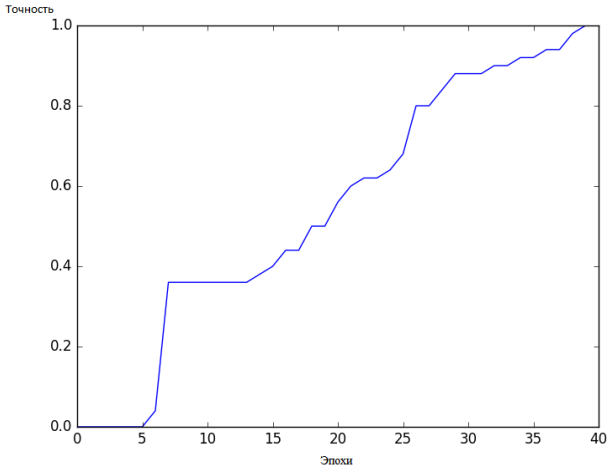

Рис. 3. Изменение точности работы сети на тестовом наборе данных на протяжении обучения на первой выборке

Рис. 3 показывает, что точность первые несколько эпох не была значительной, но внезапно выросла к седьмой эпохе, затем продолжила расти переменными скачками. Интересно, что точность сети на тренировочном наборе имеет аналогичную структуру.

Иначе выглядят результаты тестирования сети на второй выборке. Здесь не удалось достичь максимальной точности на тестовом наборе, поэтому динамика этого показателя рассмотрена в табл. 2.

Как видно из табл. 2, во время тестирования сети на второй выборке наилучшей точности в 84,9 % удалось добиться при 5 нейронах на скрытом слое. Однако следует отметить, что для этого потребовалось наибольшее число эпох. Рис. 4 демонстрирует, как изменялась точность работы сети на тестовом наборе с дальнейшим обучением.

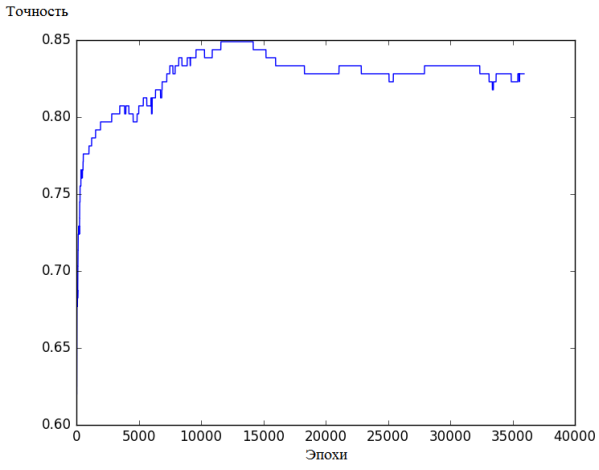

Рис. 4. Изменение точности работы сети на тестовом наборе данных на протяжении обучения на второй выборке

Из рис. 4 видно, что на протяжении нескольких тысяч эпох точность сети оставалась максимальной, но постепенно снижалась, начиная с эпохи 14187, когда сеть стала чрезмерно специализироваться. На рис. 5 видно, что точность сети на тренировочном наборе при этом только увеличивалась до последних эпох.

Таблица 2

Результаты экспериментов с различной конфигурацией сети на второй выборке

Кол-во нейронов на скрытом слое	Кол-во эпох до достижения лучшего результата	Лучшая точность на тестовом наборе
1	2876	79,6 %
2	1758	80,2 %
3	5042	79,1 %
4	7242	83,3 %
5	23184	84,9 %
6	15912	84,4 %
7	14494	84,4 %
8	7122	82,3 %
9	7350	81,8 %
10	6838	83,3 %
11	8440	82,3 %
12	7646	82,8 %
13	7388	82,3 %
14	6644	83,3 %
15	7204	83,3 %
16	7800	82,3 %
17	6538	82,8 %
18	7558	81,25 %
19	6188	81,7 %
20	8184	82,8 %

Хотя средняя квадратичная ошибка тоже снижается (рис. 6), с определенного момента сеть становится чрезмерно специализированной, и её практическая польза в классификации образцов падает.

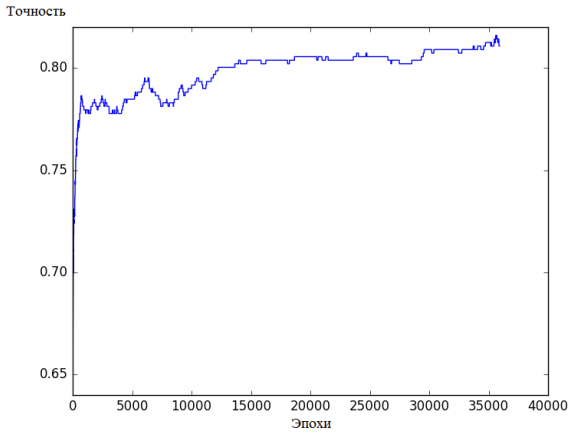

Рис. 5. Изменение точности работы сети на тренировочном наборе данных на протяжении обучения на второй выборке

Хотя в целом график на рис. 6 повторяет график падения средней квадратичной ошибки для предыдущей выборки (рис. 2), можно заметить, что он сдвинут вверх по оси J в виду большей сложности второй выборки для классификации многослойным перцептроном.

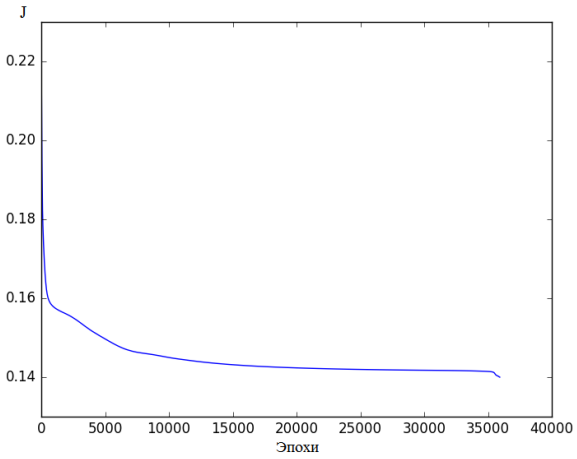

Рис. 6. Изменение средней квадратичной ошибки сети J на протяжении обучения на второй выборке

6. Обсуждение результатов

Разработанная библиотека позволила создать многослойный перцептрон, который после обучения способен классифицировать образцы с высокой точностью. Удалось достичь максимальной точности в 100 % на первой выборке, и точности 84,9 % на второй, что выше, чем результаты аналогичных исследований с многослойными перцептронами [6]. Более того, такая точность значительно выше, чем, например, применение алгоритма C4.5 на этой выборке [7].

Результаты испытаний наглядно показали опасность чрезмерной специализации сети. Чтобы этого избежать, следует регулярно отслеживать её поведение на тестовом наборе данных и вовремя останавливать обучение.

Исходя из конфигурации, на которой удалось достичь наилучшего результата на обеих рассмотренных выборках, можно предположить, что наиболее удачным количеством нейронов на скрытом слое является число, примерно равное половине – двум третьим от числа входов.

Качественных результатов можно не достичь с первого испытания в виду случайной инициализации начальных весов перцептрона. Для достижения наивысшей точности в распознавании следует многократно повторять обучение с одной и той же конфигурацией, пока не будет достигнут приемлемый результат или не будут сделаны выводы о невозможности получения приемлемого результата.

Для тестирования разработанная библиотека была откомпилирована для операционной системы Windows 7, но в виду кроссплатформенности библиотеки это не имеет принципиального значения. Библиотека может быть откомпилирована под любую другую операционную систему, на которую имеется C++ компилятор, что автоматически покрывает значительное количество широкоиспользуемых операционных систем [8]. Библиотека может быть использована в клиентском коде на C++ или любом другом языке, позволяющем использование C++ библиотек. Набор данных после классификации может быть использован в любых других целях, так что разработанная библиотека представляет собой прикладной аналитический инструмент.

Распространенные программные пакеты, позволяющие моделировать ИНС, например, Neuroph, хоть и имеют более широкий набор функционала, но позволяют генерировать код только на языке Java [9], который существенно более медленный по сравнению с кодом на C++ в виду необходимости JVM, которая создает для выполняемого кода виртуальную среду [10]. Аналогично, WEKA имеет API только для Java-приложений [11].

MATLAB, другой распространенный пакет, применяемый в том числе для анализа данных, требует от пользователя программировать на его уникальном языке программирования [12], что может быть затруднительно и требует дополнительных временных затрат на обучение.

7. Выводы

В ходе работы была разработана программная библиотека на языке C++, предоставляющая классы для конструирования, обучения и использования многослойного перцептрона, что позволяет проводить классификацию входящих образцов данных. Библиотека позволила провести эксперименты на более чем одной выборке и показала лучшие результаты в сравнении с другими программами, что свидетельствует о применимости разработки в прикладных целях.

В дальнейшем планируется расширять функционал сети добавлением других типов активационных функций и отдельного параметра импульса для погашения осцилляций коэффициентов в обучении сети обратным распространением ошибки, увеличением количества возможных скрытых слоев. Затем библиотека может быть дополнена классами для создания нейронных сетей с другой топологией, например, сверточных сетей (convolutional neural network).

Литература

1. Duda, R. O. Pattern Classification [Text] / R. O. Duda, P. E. Hart, D. G. Stork. – 2-nd ed. – N. Y.: Wiley, 2000. – P. 272–286.

2. Haykin, S. O. Neural Networks and Learning Machines [Text] / S. O. Haykin. – London; UK: Pearson, 2009. – P. 797–798.

3. Karsoliya, S. Approximating Number of Hidden layer neurons in Multiple Hidden Layer BPNN Architecture [Text] / S. Karsoliya // International Journal of Engineering Trends and Technology. – 2012. – Vol. 3, Issue 6. – P. 714–717.

4. Swain, M. An Approach for IRIS Plant Classification Using Neural Network [Text] / M. Swain // International Journal on Soft Computing. – 2012. – Vol. 3, Issue 1. – P. 79–89. doi: 10.5121/ijsc.2012.3107

5. Linear Classification with SLP [Electronic resource]. – Available at: https://grey.colorado.edu/emergent/ index.php/Linear_Classification_with_SLP

6. Shanker, M. S. Using Neural Networks to Predict the Onset of Diabetes Mellitus [Text] / M. S. Shanker // Journal of Chemical Information and Computer Sciences. – 1996. – Vol. 36, Issue 1. – P. 35–41. doi: 10.1021/ci950063e

7. Using C4.5 to predict Diabetes in Pima Indian Women [Electronic resource]. – Available at: https://datayo.wordpress.com/2015/05/13/using-c4-5/

8. An incomplete list of C++ compilers [Electronic resource]. – Available at: http://www.stroustrup.com/ compilers.html

9. Neuroph: Smart Java Apps with Neural Networks [Electronic resource]. – Available at: https://dzone.com/articles/ understanding-garbage-collection-log

10. The Java® Virtual Machine Specification [Electronic resource]. – Available at: http://docs.oracle.com/javase/ specs/jvms/se7/html/index.html

11. Documentation [Electronic resource]. – Available at: http://www.cs.waikato.ac.nz/ml/weka/documentation.html

12. MATLAB Язык технических вычислений [Электронный ресурс]. – Режим доступа: http://www.exponenta.ru/ educat/free/matlab/gs.pdf

References

1. Duda, R. O., Hart, P. E., Stork, D. G. (2000). Pattern Classification. New York: Wiley, 272–286.

2. Haykin, S. O. (2009). Neural Networks and Learning Machines. London; UK: Pearson, 797–798.

3. Karsoliya, S. (2012). Approximating Number of Hidden layer neurons in Multiple Hidden Layer BPNN Architecture. International Journal of Engineering Trends and Technology, 3 (6), 714–717.

4. Swain, M. (2012). An Approach for IRIS Plant Classification Using Neural Network. International Journal on Soft Computing, 3 (1), 79–89. doi: 10.5121/ijsc.2012.3107

5. Linear Classification with SLP. Available at: https://grey.colorado.edu/emergent/index.php/Linear_Classification_with_SLP

6. Shanker, M. S. (1996). Using Neural Networks To Predict the Onset of Diabetes Mellitus. Journal of Chemical Information and Computer Sciences, 36 (1), 35–41. doi: 10.1021/ci950063e

7. Using C4.5 to predict Diabetes in Pima Indian Women. Available at: https://datayo.wordpress.com/2015/05/ 13/using-c4-5/

8. An incomplete list of C++ compilers. Available at: http://www.stroustrup.com/compilers.html

9. Neuroph: Smart Java Apps with Neural Networks. Available at: https://dzone.com/articles/understanding-garbage-collection-log

10. The Java® Virtual Machine Specification. Available at: http://docs.oracle.com/javase/specs/jvms/se7/html/ index.html

11. Documentation. Available at: http://www.cs. waikato.ac.nz/ml/weka/documentation.html

12. MATLAB Jazyk tehnicheskih vychislenij. Available at: http://www.exponenta.ru/educat/free/matlab/gs.pdf

Рекомендовано до публікації д-р техн. наук Четвериков Г. Г.
Дата надходження рукопису 12.04.2016

Ібадов Дмитро Тарієльович, кафедра программнї інженерії, Харківський національний університет радіоелектроніки, пр. Науки, 14, м. Харків, Україна, 61166
E-mail: dibadov@gmail.com

Афанасьєва Ірина Віталіївна, кандидат технічних наук, доцент, кафедра программнї інженерії, Харківський національний університет радіоелектроніки, пр. Науки, 14, м. Харків, Україна, 61166
E-mail: iryna.afanasieva@nure.ua

УДК 614.841:543.57
DOI: 10.15587/2313-8416.2016.69601

ЗАСТОСУВАННЯ КУПРУМ(II) КАРБОНАТУ ЯК СПОСІБ ЗНИЖЕННЯ ПОЖЕЖНОЇ НЕБЕЗПЕКИ ЕПОКСИАМІННИХ КОМПОЗИЦІЙ

© О. І. Лавренюк, Б. М. Михалічко, П. В. Пастухов

Ґрунтуючись на відомостях про електронну та кристалічну будову солей купруму передбачена перспективність їх використання в якості антипіренів епоксиамінних композицій. Розроблено епоксиамінні композиції, модифіковані сполуками купруму, зокрема купрум(II) карбонатом. Досліджено вплив антипірена на процеси термоокисної деструкції епоксиамінних композицій та їх пожежну небезпеку
Ключові слова: *антипірен, купрум(II) карбонат, термоокисна деструкція, швидкість поширення полум'я, група горючості*

Being based on data about electronic and crystal structure of copper salts, their using perspective for fire retardants of epoxy-amine composites have been foreseen. Epoxy-amine composites modified by copper compounds, in particular, by copper(II) carbonate have been elaborated. Influence of fire retardant onto processes of thermal-oxidative decomposition of epoxy-amine composites and their fire hazard has been studied
Keywords: *fire retardant, copper(II) carbonate, thermal-oxidative decomposition, flame spread velocity, flammability group*

1. Вступ

Надзвичайні ситуації, зокрема пожежі, є серйозною проблемою для багатьох країн світу, в тому числі і для України. Останніми роками існує тенденція до їх збільшення. Особливу увагу привертають пожежі, які сталися внаслідок займання полімерних