Совместное использование нейросетевых технологий и деревьев решений для поиска логических закономерностей в данных

В.Н. Гридин[a], В.И. Солодовников[a]

[a]*Центр информационных технологий в проектировании РАН, 143000, ул. Маршала Бирюзова, 7а, Одинцово, Московская область, Россия*

Аннотация

Рассматриваются вопросы совместного использования нейросетевых технологий с методами логического вывода и поддержки принятия решений в задачах «интеллектуального» анализа данных. Осуществляется анализ алгоритмов поиска логических закономерностей, их достоинств и недостатков. Приводится описание комбинированных алгоритмов извлечения систем правил из обученных нейронных сетей и представления результата в виде иерархической, последовательной структуры правил типа «если-то». Рассматривается представление деревьев решения в виде фактов семантической сети.

Ключевые слова: нейронные сети; деревья решений, логический вывод; извлечение правил; «интеллектуальный» анализ данных

1. Введение

Данные являются ценным ресурсом, который хранит в себе большие потенциальные возможности по извлечению полезной аналитической информации. Поэтому, все большую актуальность приобретают задачи выявления скрытых закономерностей, выработки стратегий принятия решений, прогнозирования, что требует более подробного рассмотрения вопросов поиска логических закономерностей в задачах классификации. Особенностью алгоритмов и методов, применимых для решения задач интеллектуального анализа данных, является отсутствие ограничительных рамок априорных предположений о структуре выборки и виде распределений значений анализируемых показателей, чему наилучшим образом соответствует использование нейросетевых технологий. Это обусловлено способностью нейронных сетей к моделированию нелинейных процессов, воспроизведению чрезвычайно сложных зависимостей, адаптивностью к условиям функционирования, а главное, способностью извлекать и обобщать существенные особенности из поступающей информации. Тем самым сеть осуществляет построение правил, однако эти правила содержатся в весовых коэффициентах, функциях активации и связях между нейронами, но обычно их структура слишком сложна для восприятия и определения влияния отдельного признака на выходное значение. Нейросеть, по сути, выступает «черным ящиком», на вход которого подаются исходные данные и на выходе получается некоторый результат, однако обоснования, почему было принято именно такое решение, не предоставляется. Для решения этой проблемы предлагается совместное использование нейросетевых технологий и методов логического вывода, в частности деревьев решений, как средства поддержки принятия решений, выявления логических закономерностей и представления результата в виде иерархической структуры классифицирующих правил. А использование семантических сетей предоставляет дополнительные возможности в конструировании механизмов вывода и наглядного отображения процесса принятия решений.

2. Поиск логических закономерностей в данных

Будем понимать под логической закономерностью легко интерпретируемое правило, выделяющее из обучающей выборки достаточно много объектов какого-то одного класса и практически не выделяющее объекты остальных классов. Логические закономерности являются элементарными «строительными блоками» для широкого класса алгоритмов классификации. Правила, выражающие закономерности, формулируются на языке логических предикатов первого порядка вида:

ЕСЛИ (условие1) И (условие2) И … И (условиеN) ТО (вывод),

где условие i может быть $x_i = c_1$, $x_i < c_2$, $x_i > c_3$, $c_4 < x_i < c_5$ и т.д., x_i - переменная, c_1, c_2, c_3, c_4, c_5 – некоторые константы. Для номинативных данных используются предикаты «=» и «<>».

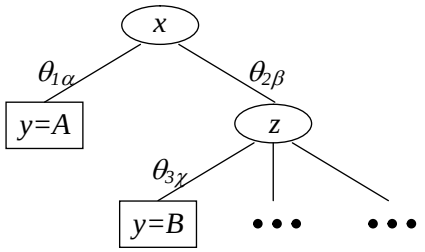
2.1. Алгоритмы ограниченного перебора

Алгоритмы ограниченного перебора используются для поиска логических закономерностей в данных, решения задач классификации и прогнозирования [1]. Основная идея этого метода заключается в анализе частоты возникновения различных комбинаций простых логических событий. На начальных этапах осуществляется поиск коротких ассоциативных цепочек, которые усложняются в процессе функционирования системы, путем добавления в них новых звенев. На основании проведенного анализа система делает заключение о полезности той или иной комбинации и, таким образом, устанавливает логические закономерности в данных. Их главным недостатком является то обстоятельство, что данный алгоритм способен за приемлемое время выдать решение только для сравнительно небольшой размерности данных.

2.2. Деревья решений

Деревья решений относятся к методам поиска логических закономерностей в данных, а также являются основным подходом, применимым в теории принятия решений, и представляют собой иерархическую структуру классифицирующих правил типа if-then (если-то), имеющих вид дерева. Их основным достоинством является простота и наглядность описания процесса принятия решения. Недостатком их использования в задаче поиска логических закономерностей является то обстоятельство, что они не способны находить наиболее полные и точные правила в данных и реализуют простейший принцип последовательного просмотра признаков и формируют лишь фрагменты закономерностей. Также при больших объемах многомерных данных алгоритмы построения деревьев решений могут выдавать очень сложную структуру деревьев, которые имеют много узлов и ветвей. А такие деревья бывает очень трудно проанализировать и понять; соответственно, правила и закономерности, выявленные из данных таким деревом, будут сложны для восприятия. К тому же ветвистое дерево, имеющее много узлов, разбивает обучающее множество на большое количество подмножеств, состоящих из малого количества объектов. Тогда как гораздо предпочтительнее иметь дерево, состоящее из малого количества узлов, которым соответствует большое количество объектов из обучающей выборки. Для решения данной проблемы часто применяются алгоритмы отсечения ветвей [2], но они не всегда могут привести к желаемому результату. Однако, методы выделения закономерностей с помощью деревьев решений позволяют находить такие связи, которые заключены не только в отдельных признаках, но и в сочетании признаков, что во многих случаях дает этим методам значительное преимущество по сравнению с классическими методами многомерного анализа.

На рисунке 1 приведен пример такого дерева решений, и соответствующий логический вывод, где $\theta1,\theta2,\theta3$ - предикаты, x, y, z – переменные, α,β,χ - константы.



$$\theta_1(x,\alpha) \rightarrow (y = A)$$
$$\theta_2(x,\beta) \cap \theta_3(z,\chi) \rightarrow (y = B)$$

Рис. 1. Пример дерева решений.

Правила, выражающие закономерности, формулируются в виде продукций: «ЕСЛИ А ТО В» или в случае множества условий: «ЕСЛИ (условие 1) ∧ (условие 2) ∧ … ∧ (условие N) ТО (Значение вершины вывода)».

Построение деревьев решений обычно осуществляется:

- на основе экспертных оценок;
- с использованием алгоритмов обработки примеров (CLS, ID3 (Interactive Dichotomizer), C4.5, CART (classification and regression trees) и др.);
- с помощью генетических алгоритмов и эволюционного программирования.

Каждый из этих подходов имеет свои достоинства и недостатки и может использоваться для решения своих конкретных задач.

2.3. Генетические алгоритмы

Наиболее сложной проблемой при поиске логических закономерностей в массивах данных является нахождение элементарных событий, представляющих термы условной части (части ЕСЛИ). В настоящее время для решения этой проблемы все чаще применяются генетические алгоритмы (ГА), к которым можно отнести алгоритмы «команды пожарных» (Bucket-Brigade), REGAL, G-NET, HIDER, SIAO1 и ряд других. Однако они не лишены ряда недостатков: фиксированный набор правил и их длина, а также точность и полнота в большинстве из них не учитывается.

2.4. Нейросетевые методы

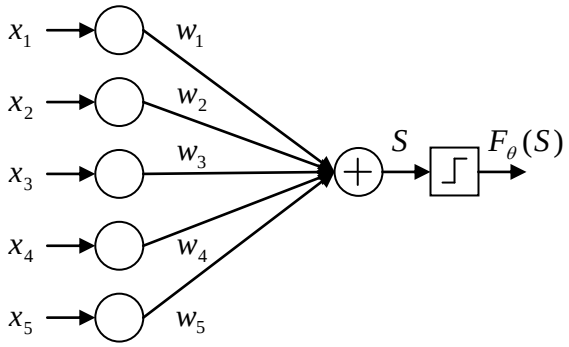
Использование подхода, основанного на нейросетевых технологиях обработки данных, обусловлено способностью нейронных сетей к моделированию нелинейных процессов, воспроизведению чрезвычайно сложных зависимостей, адаптивности к условиям функционирования, работе с зашумленными данными и отсутствием априорной информации. А главное, они способны обучаться на основе опыта, обобщать предыдущие прецеденты на новые случаи и извлекать существенные особенности из поступающей информации. Тем самым сеть осуществляет построение правил, однако эти правила содержатся в весовых коэффициентах, функциях активации и связях между нейронами, но обычно их структура слишком сложна для восприятия. Более того, в многослойной сети, эти параметры могут представлять собой нелинейные, немонотонной отношения между входными и целевыми значениями. Таким образом, как правило, не представляется возможным отделить влияние определенного признака на целевое значение, потому что этот эффект может быть опосредован значениями других параметров. Нейросеть, по сути, выступает «черным ящиком», на вход

которого подаются исходные данные и на выходе получается некоторый результат, однако обоснования, почему было принято именно такое решение, не предоставляется.

3. Выделение правил из обученной нейронной сети

Пусть задача состоит в классификации некоторого набора данных с помощью персептрона и последующего анализа полученной сети с целью нахождения классифицирующих правил, характеризующий каждый из классов.

Сначала рассмотрим данную задачу на примере однослойного персептрона, в котором пять булевых нейронов на входе и один на выходе. Данная сеть может быть в точности интерпретирована конечным числом правил «если-то», так как для нее определено конечное число возможных входных векторов.



$$S = \sum_i w_i \cdot x_i$$

$$y = F_\theta(S) = \begin{cases} 1 & if\ S > \theta \\ 0 & else \end{cases}$$

Рис. 2. Однослойный персептрон с пятью булевыми входами и одним выходом.

Пусть веса принимают значения: $w_1 = 6$, $w_2 = 4$, $w_3 = 4$, $w_4 = 0$, $w_5 = -4$, а порог $\theta = 9$. В этом случае из сети можно извлечь следующий набор правил:

$$x_1 \wedge x_2 \wedge x_3 \rightarrow y$$

$$x_1 \wedge x_2 \wedge \neg x_5 \rightarrow y$$

$$x_1 \wedge x_3 \wedge \neg x_5 \rightarrow y$$

Таким образом, процедура принятия решений заключается в предсказании значения $y = true$, если активация выходного нейрона равна 1, и $y = false$, когда активация равна 0.

Вообще говоря, можно выделить два подхода к извлечению правил из многослойных нейронных сетей [3]. Первый подход заключается в извлечении набора глобальных правил, которые характеризуют классы на выходе непосредственно через значения входных параметров. Альтернативой является извлечение локальных правил, разделяя многослойную сеть на совокупность однослойных сетей. Каждое извлекаемое локальное правило характеризует отдельный скрытый или выходной нейрон с учетом элементов, которые имеют с ним взвешенные соединения. Затем правила объединяются в набор, который определяет поведение всей сети в целом. Локальный подход проиллюстрирован на рисунке 3.

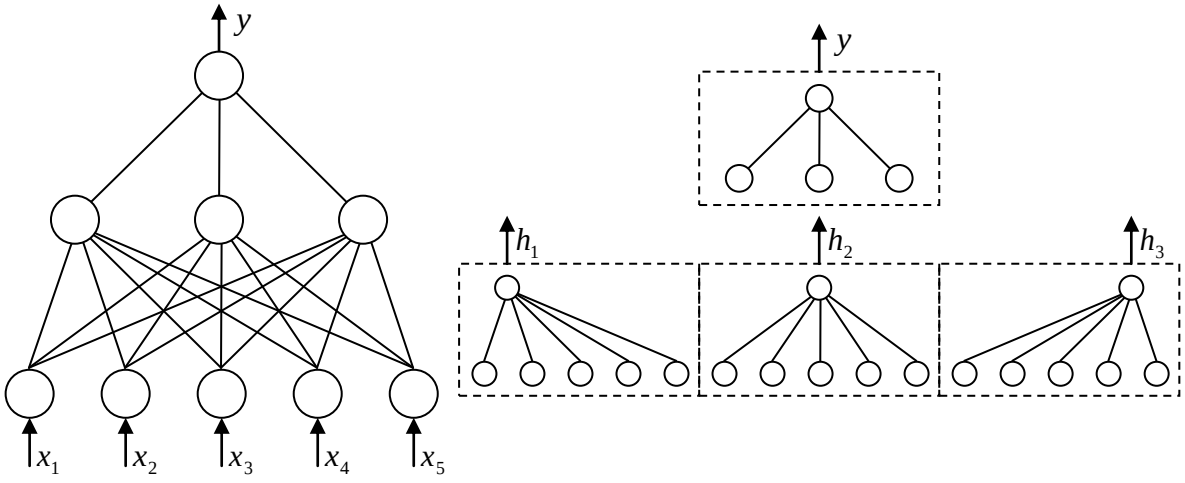


Рис. 3. Локальный подход к извлечению правил. Многослойная нейронная сеть разделяется на набор однослойных. Для описания каждой из составных частей извлекаются правила, которые в дальнейшем объединяются в набор, характеризующий многослойную сеть.

Рассмотрим задачу извлечения правил в более общем виде.

Пусть X обозначает набор из n свойств $X_1, X_2, ..., X_n$, а $\{x_i\}$ - множество возможных значений, которое может принимать свойство X_i. Обозначим через C множество классов $c_1, c_2, ..., c_m$. Для обучающей выборки известны ассоциированные пары векторов входных и выходных значений $(x_1, ..., x_n, c_j)$, где $c_j \in C$.

3.1. Локальный подход к извлечению правил

Одним из алгоритмов извлечения правил из нейронных сетей, обученных решению задачи классификации, является метод NeuroRule [4]. Данный алгоритм включает три основных этапа:

Этап 1. Обучение нейронной сети.

На первом этапе двухслойный персептрон обучается вплоть до получения достаточной точности классификации. В первоначальный момент времени выбирается большое число промежуточных нейронов и после обучения излишние нейроны и связи отбрасываются.

Этап 2. Прореживание нейронной сети.

Обученная нейронная сеть содержит все возможные связи между входными нейронами и нейронами скрытого слоя, а также между последними и выходными нейронами. Полное число этих связей обычно столь велико, что из анализа их значений невозможно извлечь обозримые для пользователя классифицирующие правила. Прореживание заключается в удалении излишних связей и нейронов, не приводящем к увеличению ошибки классификации сетью. Результирующая сеть обычно содержит немного нейронов и связей между ними, и функционирование такой сети поддается исследованию.

Этап 3. Извлечение правил.

На этом этапе из прореженной нейронной сети извлекаются правила, имеющие форму «если ($x_1 \Theta q_1$) и ($x_2 \Theta q_2$) и ... и ($x_n \Theta q_n$), то c_j », где $q_1,\ldots, q_n$ - константы, Θ - оператор отношения ($=, \geq, \leq, ><$). Для этого проводят подготовку к извлечению правил, которая заключается в кодировании непрерывных величин, как на входе, так и внутри сети. Осуществляется кодирование признаков классифицируемых объектов, если они представляют собой непрерывные величины. Для их представления можно использовать бинарные нейроны и принцип кодирования типа термометр. Значения, которые принимают нейроны скрытого слоя кластеризуются и заменяются значениями, определяющими центры кластеров. Число таких кластеров выбирается небольшим. После такой дискретизации активностей промежуточных нейронов производится проверка точности классификации объектов сетью. Если она остается приемлемой, то подготовка к извлечению правил заканчивается. Далее осуществляется извлечение правил, при этом движение по сети происходит от классифицирующих выходных нейронов к входам сети. Предполагается, что эти правила достаточно очевидны при проверке и легко применяются к большим базам данных.

Однако, данный алгоритм устанавливает довольно жесткие ограничения на архитектуру нейросети, числу элементов, связей и виду функций активации. Так для промежуточных нейронов используется гиперболический тангенс и их состояния изменяются в интервале [-1,1], а для выходных нейронов применяется функция Ферми с интервалом состояний [0,1].

3.2. Извлечение глобальных правил

К недостаткам большинства алгоритмов извлечения правил можно отнести отсутствие универсальности и масштабируемости. В связи с этим, наибольший интерес представляет алгоритм TREPAN [5], который лишен этих недостатков и не предъявляет никаких требований к архитектуре сети, входным и выходным значениям, алгоритму обучения и т.д. Данный подход осуществляет построение дерева решений на основе знаний, заложенных в обученную нейросеть, причем достаточно того, что сеть является неким «черным ящиком» или «оракулом», которому можно задавать вопросы и получать от него ответы. Более того, алгоритм является достаточно универсальным и может применяться к широкому кругу других обученных классификаторов. Он также хорошо масштабируется и не чувствителен к размерности пространства входных признаков и размеру сети.

Алгоритм построения дерева решения, аппроксимирующего работу обученной нейронной сети состоит из двух этапов.

Предварительный этап:

1. Построить и обучить нейронную сеть, которая в дальнейшем будет выступать в роли «Оракула».
2. Инициализировать корень дерева R в виде листа.
3. Использовать все обучающее множество примеров S для конструирования модели M_R распределения входных векторов, достигающих узла R. Вычислить значение q = max(0, minSamples - |S|), где minSamples – минимальное число обучающих примеров, используемое в каждом узле дерева, S – текущая обучающая выборка (|S|-объем обучающей выборки). Таким образом, q – количество дополнительных примеров, которые необходимо сгенерировать.
4. На основе оценки распределения признаков из S, случайным образом, генерируются q новых обучающих примеров. $query_R$ - множество из q примеров, генерируемых моделью M_R.
5. Использовать нейронную сеть «Оракула» для классификации, как новых $query_R$, так и старых примеров из множества S к тому или иному классу. Для каждого вектора признаков $x \in (S \cup query_R)$ выставить метку класса $x = Oracle(x)$.
6. Инициализировать очередь $Queue$, поместив в нее набор $\langle R, S, query_R, \{empty\_constr\} \rangle$.

Основной этап:

7. Взять очередной набор $\langle N, S_N, query_N, constr_N \rangle$ из начала очереди $Queue$, где N - узел дерева, S_N - обучающая выборка в узле N, $constr_N$ - набор ограничений на определенные признаки обучающих примеров для достижения узла N.

8. Использовать $F, S_N, query_N$ для конструирования в узле N разветвления T.
   Здесь F - функция, оценивающая узел N. Она имеет вид $F(N) = R(N) \cdot (1 - f(N))$, где $R(N)$ - вероятность достижения узла N примером, а $f(N)$ - оценка правильности обработки этих примеров деревом. Таким образом, выбирается наилучший узел, разветвление которого оказывает наибольшее влияние на точность классификации генерируемого дерева.
   Разделение примеров, достигающих данный внутренний узел дерева, осуществляется в зависимости от $m-of-n$ теста [5,6]. Такой тест считается пройденным, когда выполняются, по меньшей мере, m из n условий.
   С другой стороны, возможно расщепление множества S как в обычном алгоритме построения дерева решений.
9. Для каждой дуги t разветвления T создать узлы следующего поколения:
   a. Создать C - новый дочерний узел по отношению к N.
   b. $constr_C = constr_N \cup \{T = t\}$ - добавить ограничение с дуги t.
   c. Сформировать $S_C =$ примеры из множества S_N, которые удовлетворяют условию на дуге t.
   d. Сконструировать модель M_C распределения примеров, достигающих узла C.
      Подсчитать значение q = max(0, minSamples - $|S_C|$), т.е. количество примеров, которые необходимо сгенерировать
   e. На основе оценки распределения признаков из S_C и значения ограничений $constr_C$, случайным образом, сгенерировать q новых обучающих примеров $query_C$ – множество из q примеров, сгенерированных моделью M_C и ограничением $constr_C$
   f. Использовать нейронную сеть «Оракула» для классификации новых примеров $x \in query_C$ и выставить метку класса $x = Oracle(x)$.
   g. Изначально предполагается, что узел С является листом. Использовать S_C и $query_C$ для определения метки класса для С.
   h. Проверить необходимость дальнейшего расщепления узла С. Если локальный критерий остановки не удовлетворен, то поместить набор $< C, S_C, query_C, constr_C >$ в очередь $Queue$. Локальным критерием в данном случае выступает величина, которая характеризует вероятность, что в данном узле встречаются экземпляры одного класса.
10. Если очередь $Queue$ не пуста и не выполнен глобальный критерий остановки, то перейти к шагу 7, иначе вернуть дерево с корнем R.

В качестве глобального критерия завершения алгоритма используется максимальный размер дерева и общую оценку качества классификации примеров деревом.

Основное преимуществом данного подхода заключается в обобщающей способности искусственных нейронных сетей, которая позволяет получать более простые деревья решений. К тому же, использование такого «Оракула» позволяет компенсировать недостаток данных, наблюдающийся при построении деревьев решений на нижних уровнях.

Таким образом, возможно извлечение структурированных знаний не только из чрезвычайно упрощенных нейронных сетей, но и из произвольных классификаторов, что делает возможным применение данного алгоритма в широком круге практических задач.

4. Представление правил в виде семантической сети

Простая семантическая сеть, называемая иногда вычислительной семантической сетью, представляет собой фактически двудольный граф.

Пусть задано конечное множество A = {A1, . . ., Ar}, называемых атрибутами, и конечное множество R = {R1, . . ., Rn} отношений. Схемой или интенсионалом отношения Ri ($i = \overline{1,n}$) называют набор пар:

INT(Ri) = { . . ., [Aj, DOM(Aj)], . . .},

где Ri – имя отношения, DOM(Aj) - домен Aj ($j = \overline{1,r}$), т.е. множество значений атрибута Aj отношения Ri.

Объединение всех доменов называется базовым множеством модели или множеством объектов, на которых задаются отношения R.

Экстенсионалом отношения Ri называют множество

EXT(Ri) = {F1, . . ., Fp},

где Fk ($k = \overline{1,p}$) – факт отношения Ri. Факт задается совокупностью пар атрибут – значение, называемых атрибутивными парами. Под фактом понимается конкретизация определенного отношения между указанными объектами. В графической интерпретации факт – это подграф семантической сети, имеющий звездообразную структуру. Корень подграфа – вершина предикатного типа, помеченная уникальной меткой, включающей имя соответствующего отношения. Из вершины факта выходят ребра, помеченные именами атрибутов данного факта, ведущие в вершины базового множества, которые являются значениями этих атрибутов.

Стоит отметить, что семантическая сеть может быть представлена как хранилище фактов, выведенных в результате обработки деревьев решений, т.е. осуществить преобразования дерева решений в семантическую сеть. В этом случае каждый факт представляется в виде уже готового варианта вывода. Это дает дополнительные возможности для анализа. Например, обычными средствами баз данных можно выделить факты, относящиеся даже к разным отношениям,

обладающие одинаковыми атрибутами и значениями их характеризующими. Для хранения в базе данных семантических сетей, точнее их экстенсионалов, можно использовать таблицу вида:

Таблица 1. Таблица экстенсионалов.

Имя поля	Тип данных	Свойства поля
CodeValue	Числовой	Ключевое поле
FactMark	Числовой	Метка факта
FactAttributes	Текстовый	Атрибут факта
AttributeValue	Текстовый	Значение атрибута

Таким образом, будет получено представление дерева решений в виде таблицы фактов, где для каждого значения переменной вывода присутствует своя запись значений атрибутов факта. Поскольку существует отображение одного представления в другое, эти представления с формальной точки зрения тождественны.

В семантической сети можно осуществить выводы, которые далеко не очевидны для дерева решений. Возьмем такой простейший пример. Пусть в дереве решений определен вывод установления родительских отношений в первом поколении. Необходимо определить родительские отношения во втором поколении. Решение этой задачи для семантической сети очевидно. Необходимо выделить факты родительских отношений. Удалить из интенсионала объекты, не соответствующие родителям и повторить процесс выделения фактов родительских отношений.

Это означает, что в семантической сети возможно введение средств построения функциональных зависимостей подобно тому, как это сделано в языках функционального программирования таких как, например, ЛИСП [7]. Можно рассматривать факт, как список атомов, каждому из которых ставится в соответствие значение либо непосредственно, либо в процессе вывода. Если ввести операции манипулирования над такими списками, появляется возможность создавать и модифицировать деревья решений формальными методами.

Таким образом, представляется целесообразным сочетать в единой системе представление в виде деревьев решений и класса семантических сетей, что дает возможность наглядного отображения процесса принятия решений и дополнительные возможности в конструировании механизмов вывода.

5. Заключение

В работе рассмотрены вопросы поиска логических закономерностей в задачах классификации. Предложено совместное использование нейросетевых технологий и методов логического вывода, в частности деревьев решений, как средства выявления логических закономерностей и представления результата в виде иерархической структуры классифицирующих правил. Выделены два основных подхода. Первый заключается в извлечении локальных правил, где многослойная сеть разделяется на совокупность однослойных. Каждое локальное правило характеризует отдельный скрытый или выходной нейрон с учетом элементов, которые имеют с ним взвешенные соединения. Затем правила объединяются в набор, который определяет поведение всей сети в целом. Однако, данный подход зачастую устанавливает довольно жесткие ограничения на архитектуру сети, число элементов, связей и вид функций активации, что отрицательно сказывается на универсальности и масштабируемости.

Альтернативой является извлечение набора глобальных правил, которые характеризуют классы на выходе непосредственно через значения входных параметров. В рамках данного подхода разработана модификация алгоритма построения деревьев решений на основе обученных нейронных сетей, которая не предъявляет никаких требований к архитектуре, алгоритму обучения, входным и выходным значениям и другим параметрам сети. Построение дерева осуществляется на основе знаний, заложенных в обученную нейросеть, причем достаточно того, что сеть является неким «черным ящиком», которому можно задавать вопросы и получать от него ответы. Основное преимущество заключается в обобщающей способности нейронной сети, что позволяет получать более простые деревья решений, а также, в случае необходимости, компенсировать недостаток исходных данных. Более того, алгоритм является достаточно универсальным, хорошо масштабируемым и не чувствительным к размерности пространства входных признаков и размеру сети. Данное обстоятельство приобретает особое значение в свете бурного развития технологии глубинного обучения (Deep learning). Таким образом, возможно извлечение структурированных знаний не только из чрезвычайно упрощенных нейронных сетей, но и из произвольных классификаторов, что делает возможным применение данного алгоритма в широком круге практических задач.

Дополнительно разработан алгоритм преобразования уже сформированных деревьев решений в семантические сети в виде двудольного графа, что обеспечивает представление дерева решений в виде таблицы фактов и позволяет осуществлять быстрый поиск по известным атрибутам.

Внедрение средств автоматизации в системы интеллектуального анализа данных способно сократить сроки, повысить качество и эффективность принимаемых решений.

Благодарности

Работа выполняется при финансовой поддержке РФФИ в рамках проекта 15-07-01117а.

Литература

[1] Дюк, В. Data Mining: учебный курс / В. Дюк, А. Самойленко - СПб: Питер, 2001. - 368 с.

[2] Сайт компании BaseGroup Labs, Деревья решений - общие принципы работы [Электронный ресурс]. - Режим доступа: https://basegroup.ru/community/articles/description (10.01.2017)

[3] Гридин, В.Н. Построение деревьев решений и извлечение правил из обученных нейронных сетей / В.Н. Гридин, В.И. Солодовников, И.А. Евдокимов, С.В. Филиппков // Искусственный интеллект и принятие решений - 2013. - № 4. - С. 26-33.

[4] Ежов, А.А. Нейрокомпьютинг и его применение в экономике и бизнесе / А.А. Ежов, С.А. Шумский – М.: МИФИ, 1998. - 224 с.

[5] Craven, M.W. Extracting tree-structured representations of trained networks / M.W. Craven, J.W. Shavlik // Advances in Neural Information Processing Systems. MIT Press, Cambridge, MA. – 1996. – Vol. 8. - P. 24-30.

[6] Murphy, P.M. ID2-of-3: Constructive induction of M-of-N concepts for discriminators in decision trees / P.M. Murphy, M.J. Pazzani // In Proceedings of the Eighth International Machine Learning Workshop, Evanston, IL. - 1991. - P. 183-187.

[7] Хювенен, Э. Мир ЛИСПА. Методы и системы программирования / Э. Хювенен, И. Сеппянен – М., Мир, 1990. - 320 с.