Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего профессионального образования
«Уральский федеральный университет имени первого Президента России
Б.Н. Ельцина»
Кафедра интеллектуальных информационных технологий

Оценка работы _____

Руководитель от УрФУ _____

ПОСТРОЕНИЕ ПРИЗНАКОВ ДЛЯ КЛАССИФИКАЦИИ МЕДИЦИНСКИХ ИЗОБРАЖЕНИЙ ПОСРЕДСТВОМ ГЕНЕТИЧЕСКОГО ПРОГРАММИРОВАНИЯ

ОТЧЕТ
по преддипломной практике

Руководитель от предприятия _____

Студент _____

Группа _____

Екатеринбург
2017

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

В последнее время одним из актуальных направлений развития компьютерных технологий в медицине становится обработка и анализ цифровых изображений. Медицинские изображения – это структурно-функциональный образ органов человека, предназначенный для диагностики заболеваний и изучения анатомо-физиологической картины организма. Распознавание патологических процессов является одной из наиболее важных задач обработки и анализа медицинских изображений.

Классификация заключается в прогнозировании значения категориального атрибута (класса) на основе значений других атрибутов. Цель классификации – взять объект, оцениваемый набором признаков, и назначить его одному дискретному классу.

Для решения задачи классификации выполняется определенный вид предварительной обработки, известной как преобразование представления, которое, учитывая исходный вектор признаков F_0 и обучающее множество L, состоит в создании представления F, полученного из F_0, которое максимизирует некоторый критерий, и, по крайней мере, так же хорошо, как F_0 по отношению к этому критерию.

Разработка хорошего пространства признаков является предпосылкой для достижения высокой производительности в любой задаче машинного обучения. Однако часто неясно, каким должно быть оптимальное представление признака. В результате общий подход заключается в том, чтобы использовать все доступные атрибуты в качестве признаков и оставить проблему идентификации полезных наборов признаков модели обучения. Такой упрощенный подход не всегда работает хорошо.

Для решения проблем нерелевантности и взаимодействия признаков используются методы построения признаков. Конструирование признака включает в себя преобразование заданного набора входных признаков для

создания нового набора более мощных признаков, которые затем используются для прогнозирования.

В данной работе применяется генетическое программирование для автоматического конструирования признаков с целью повышения различающей эффективности классификатора.

Актуальность выбранной темы обосновывается тем, что проблема классификации патологических процессов по данным медицинских изображений не может быть правильно решена, если важные взаимодействия и отношения между оригинальными признаками, не принимаются во внимание. Сейчас в большинстве случаев выделение признаков осуществляется человеком на основе знаний исследователя, опыта и / или интуиции.

Проблема. Эффективность работы классификатора сильно зависит от входного множества признаков. Как выбрать оптимальное множество признаков для классификатора?

Цель – нахождение и отработка методики построения признаков для решения задачи классификации.

Для достижения поставленной цели необходимо выполнить следующие задачи:

– изучить соответствующую литературу;

– разработать и реализовать алгоритм построения признаков;

– оценить эффективность работы алгоритма и сравнить с классической классификацией;

– обобщить полученные результаты и сделать соответствующие выводы.

Объектом исследования выступает классификация изображений. Предметом исследования является алгоритм построения признаков для классификации.

1 Постановка задачи

Конструирование признаков может применяться для достижения двух различных целей: уменьшения размерности данных и улучшения показателей прогнозирования. В данной работе будет рассматриваться использование построения признаков для увеличения эффективности классификации.

1.1 Формальные определения

Пусть:

1. $x \in X$, где X – область входных значений.
2. $y \in Y$, где Y – область выходных значений.
3. S – набор обучающих примеров таких, что
$$S = \{(x_i, y_i)\}_{i=1\ldots m}.$$
4. S' – набор тестовых примеров таких, что
$$S' = \{(x_i, y_i)\}_{i=1\ldots m'}.$$
5. Err_h будет ошибкой гипотезы h по сравнению с истинной основной гипотезой h_T.

Мы можем рассматривать каждый x как вектор фиксированной длины значений признаков в исходном пространстве признаков, т.е.

$$x = < f_1, f_2, f_3, \ldots, f_n >,$$

где $f_i \in F_0 \; \forall \; i = 1\ldots n$ и $n = |F_0|$.

Цель любой обучающей машины – узнать предиктор $h : X -> Y$ из S, с маленькой ошибкой Err_h. В парадигме построения признака каждый исходный вектор признаков x преобразуется в новый вектор признаков

$$x' = \varphi(x) = < \varphi_1(x), \varphi_2(x), \varphi_3(x), \ldots, \varphi_N(x) >,$$

где $\varphi_i \in F_T \; \forall \; i = 1\ldots N$ и $N = |F_T|$.

Каждое преобразованное значение признака $\varphi_i(x)$ получается путем оценки некоторой функции по всем исходным f_i. Мы хотим вывести гипотезу h': $\varphi(x) -> Y$, предполагая, что ее истинная ошибка $Err_{h'}$ меньше Err_h. В большинстве практических сценариев Err_h и $Err_{h'}$ будут вычисляться путем измерения производительности h и h' на тестовом множестве S'.

1.2 Требования к системе

Необходимо разработать алгоритм, который решает задачу построения множества признаков для эффективной классификации медицинских изображений методом генетического программирования. Создаваемый алгоритм должен удовлетворять следующим критериям:

– на основе входных данных в виде множества оригинальных признаков, обучающего и тестового множеств исходных медицинских изображений, набора классификаторов, а также максимально возможной погрешности выдавать результат в виде множеств сконструированных признаков, которые обеспечивают точность классификации не меньше заданной.

– иметь возможность изменения параметров генетического программирования – мощность начальной популяции, максимальная глубина дерева, вероятности скрещивания, репродукции и мутации;

– быть удобной в использовании, т.е. предоставлять промежуточные и конечные результаты работы в удобном для пользователя виде.

2 Обзор литературы

2.1 Общее описание построения признаков

Концептуально любой метод построения признака можно рассматривать как выполнение следующих действий:

1) Выбор начального пространства признаков F_0 (ручное построение признаков).
2) Преобразование F_0 для построения нового пространства признаков F_N (преобразование признаков).
3) Выбор подмножества признаков F_i из F_N (выбор признака).
    a. Определение полезности F_i для задачи классификации.
    b. Если некоторые критерии завершения достигнуты:
        i. Вернитесь к шагу 3.
    c. Иначе множество $F_T = F_i$.
4) F_T – это сконструированное пространство признаков.

Исходное пространство признаков F_0 состоит из признаков, созданных вручную, которые часто кодируют некоторые базовые знания области. Различные методы построения признаков отличаются тем, как они реализуют каждый из этих этапов. Ясно, что тремя важными аспектами любого метода построения признаков являются: метод трансформации, метод выбора подмножества признаков F_i и критерий полезности для подмножества признаков.

2.1.1 Преобразование признаков

Общим подходом к созданию преобразованных признаков является применение набора операторов (например, {+, –, *}) к исходным значениям признаков. Выбор операторов основан на знании области и типе признаков. К числу наиболее часто используемых операторов относятся:

1) Логические функции: конъюнкция, дизъюнкция, отрицание и т. д.
2) Номинальные характеристики: декартово произведение и т. д.

3) Численные характеристики: минимум, максимум, сложение, вычитание, умножение, деление, среднее, эквивалентность, неравенство и т. д.

Помимо этого, гиперплоскости, логические правила и битовые строки также могут использоваться для создания новых пространств признаков. Операторы обычно применяются итеративно. Поэтому каждый новый признак $\varphi_i \in F_N$ можно представить, используя дерево операторов и исходные значения признаков, как показано на рисунке 1.

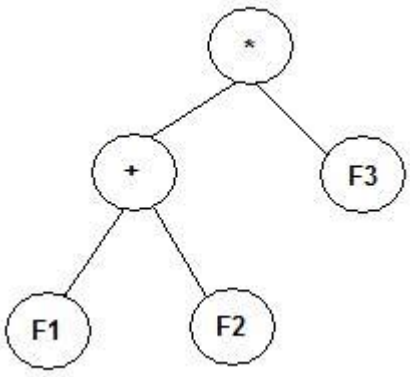

Рисунок 1 – Древовидное представление сконструированного признака $\varphi_i \in F_N$

2.1.2 Выбор признаков

Выбор признаков является важным шагом в процессе построения признака. Так как преобразованное пространство признаков F_N велико, нам нужно выбрать подмножество F_T из F_N. Проблема выбора оптимального подмножества является NP трудной, и методы обычно выполняют какой-то неоптимальный жадный поиск. Используемые критерии для измерения полезности пространства признаков F_i часто включают в себя получение информации, коэффициент корреляции, точность предсказания на некотором множестве проверки и т. д.

В литературе было представлено множество различных методов отбора (см. [4] и [3]). Мы можем свободно классифицировать эти методы по двум категориям: фильтры и обертки [7].

Фильтрующие методы выбирают подмножества признаков независимо от предиктора. Они, по существу, действуют как этап предварительной

обработки данных, прежде чем обучить предиктор. В эту категорию входят подходы с переменным ранжированием, которые включают ранжирование отдельных признаков с использованием теоретико-информационных или корреляционных критериев, а затем построение подмножества с высокими показателями выигрыша. Фильтры имеют преимущество в том, что они быстрее, чем обертки. Более того, они, как правило, обеспечивают общий порядок построения признаков, не настроенный на конкретный метод обучения. Однако недостатком является то, что выбранное подмножество может быть не самым подходящим для конкретного классификатора.

Оберточные методы выбора признаков используют метод обучения, который будет использоваться для прогнозирования как черный ящик для выбора подмножеств признаков. Эти методы обычно делят обучающий набор на набор обучения и проверки (набор тестов является отдельным). Для любого заданного подмножества признаков предиктор обучается набору обучения и тестируется на наборе проверки. Точность прогнозирования на наборе проверки рассматривается как оценка подмножества признаков. Таким образом, мы в конечном счете хотели бы выбрать подмножество с наивысшей оценкой. Из-за повторяющихся циклов обучения и тестирования для каждого подмножества признаков, обертки имеют тенденцию быть намного более вычислительно затратными по сравнению с фильтрами. Обычно цель состоит в том, чтобы пропустить пространство признаков таким образом, чтобы число проверяемых подмножеств было сведено к минимуму. Очевидным преимуществом является то, что выбранное подмножество настроено на предиктор.

2.2 Обзор аналогов

Задача построения соответствующих признаков часто очень специфична для области применения и трудоемка. Таким образом, создание автоматизированных методов построения признаков, требующих

минимальных усилий пользователя, является сложной задачей. В частности, необходимы методы, которые:

1) Создают набор признаков, которые помогут улучшить точность прогнозирования.
2) Вычислительно эффективны.
3) Являются обобщаемыми для разных классификаторов.
4) Позволяют легко добавлять знания области.

Был предложен ряд различных методов. В следующих подразделах мы классифицируем эти методы на основе методик, которые они используют для определения и поиска пространства признаков. Ранние методы в основном базировались на деревьях решений, в то время как последние подходы были в большей степени сосредоточены на индуктивном логическом программировании и генетическом программировании. Методы на основе генетического программирования являются гибкими в операторах, которые они могут использовать, в то время как основанные на индуктивном логическом программировании методы позволяют легко внедрять знания из разных источников.

2.2.1 Деревья решений

Один из ранних алгоритмов построения признаков принадлежит Пагалло [15], создателю FRINGE, который адаптивно увеличивал первоначальный набор атрибутов для изучения концепции дизъюнктивной нормальной формы (ДНФ). На каждой итерации новые признаки строятся путем комбинирования пар признаков в существующем пространстве признаков с использованием операторов отрицания и конъюнкции. Поскольку данные операторы являются полным набором булевых операторов, общее пространство признаков состоит из всех булевых функций исходных признаков. Чтобы справиться с проблемой чрезвычайно большого пространства новых признаков, FRINGE объединяет только пары признаков, которые появляются на краю каждой из положительных ветвей дерева

решений. Процесс создания признака повторяется до тех пор, пока не будут созданы новые признаки.

CITRE [14] и Fringe [22] – два других алгоритма построения признака на основе деревьев решений. Алгоритмы используют конъюнкции и дизъюнкции для объединения различных операндов, таких как корень (выбирает первые два признака каждой положительной ветви), границу (подобно FRINGE), корневую границу (комбинация корня и границы), смежную (выбирает все соседние пары вдоль каждой ветви) и все (всё вышеперечисленное).

Проблема с алгоритмами на основе деревьев решений заключается в том, что с тех пор, как новые признаки добавляются в пространство признаков на каждой итерации, количество входных признаков, которые необходимо передать в алгоритм построения дерева решений, становится очень большим, делая процесс вычислительно неэффективным. В результате в каждой итерации некоторые низко оцениваемые признаки отбрасываются. Помимо использования обычных методов обрезки дерева решений, все признаки, которые не использовались при построении дерева решений, также могут быть исключены.

Все методы, рассмотренные ранее, использовали только логические операторы для генерации признаков. Чтобы разработать более гибкий подход, Маркович и Розенштейн [13] представили FICUS, общую структуру построения признаков. Их подход обеспечивает набор функций (операторов) конструктора вместе с исходным набором признаков и примерами для алгоритма построения признаков. Затем алгоритм обогащает исходное пространство признаков добавлением дополнительных перспективных признаков. Функции конструктора могут быть либо одним, либо несколькими обычно используемыми операторами, либо могут поставляться некоторым экспертом предметной области.

Вход в FICUS задается с использованием «языка спецификаций признаков» (ЯСП). Пользователь может предоставить информацию о типе (например, номинальном, непрерывном и т.д.), области и диапазоне основных

признаков и функций конструктора. Также пользователь может дополнительно указать набор булевых ограничений на тип признаков, которые могут быть сгенерированы или использованы с определенными функциями. Таким образом, новое пространство признаков представляет собой набор всех признаков, которые могут быть сгенерированы на основе ЯСП. Обход пространства поиска осуществляется с помощью четырех разных операторов:

1) Композиция. Один или два признака используются в качестве входных данных. На их основе вычисляется новый набор признаков с использованием всех допустимых функций.

2) Вставка. На вход подаются два признака. Новый признак получается путем вставки одного признака в другой.

3) Замена. Из двух начальных признаков формируется новый признак с помощью замены какого-либо компонента одного признака на компонент другого признака.

4) Интервал. Из одного исходного признака получаются новые логические признаки, проверяющие, лежит ли начальный признак в каком-либо определенном диапазоне.

Стратегия поиска является вариантом лучевого поиска. На каждом шаге поддерживаются два набора признаков: текущий набор признаков и предыдущий набор признаков, который создал текущий набор через применение четырех указанных выше операторов. Сохранение предыдущего набора позволяет системе выполнить один уровень обратного отслеживания. Полезность набора признаков вычисляется на основе размера и сложности дерева решений, которое генерируется по множеству примеров. Полезность отдельных признаков в наборе вычисляется с использованием критериев разделения (получения информации). На каждой итерации используется лучший набор признаков для создания нового набора признаков. Авторы показали, что их метод достиг значительного прироста производительности в разных областях и классификаторах.

Обладая высокой гибкостью, с одной стороны, у FICUS было два потенциальных недостатка, с другой стороны. Во-первых, их критерии выбора подмножества признаков не учитывали взаимодействия признаков, что приводило к несколько узкому поиску в пространстве признаков. Во-вторых, как обсуждалось ранее, при заданной проблеме выбор операторов часто неясен, что затрудняет пользователям предоставление правильного набора функций конструктора.

2.2.2 Индуктивное логическое программирование

Индуктивное логическое программирование (ИЛП) используется для разработки описаний предикатов из примеров и базовых знаний. Методы построения признаков на основе ИЛП могут обеспечить обобщенную структуру для включения знаний области в процесс создания признаков. Первое использование предикатов первого порядка в качестве признаков было применено в программе LINUS [11]. В последующей литературе проблема идентификации хороших признаков с представлением первого порядка была рассмотрена в пропозициональных (основанных на характеристиках) подходах [9].

В работах [19, 20] использовался подход, основанный на ИЛП, для повышения точности прогнозирования в задаче определения значения слова. Основная идея заключается в применении двухэтапного подхода:

1) Конструирование признака. ИЛП используется для изучения нового набора конъюнктивных признаков.

2) Выбор признака. Выбор подмножества признаков на основе их полезности в процессе прогнозирования.

Их система изучает предложения вида

$$h_i: Class\ (x, c) < -\varphi_i\ (x),$$

где $\varphi_i(x)$: $X -> 0, 1$.

φ_i – это конъюнкция над некоторыми значениями признака экземпляра *x* и вычисляется как TRUE (1), если конъюнкцией является, иначе – FALSE (0).

Предложения h_i можно рассматривать как правила вида: «Если какой-либо признак $\varphi(x) = 1$, то экземпляр x должен принадлежать классу c. Когда такие правила выводятся из набора примеров, пространство признаков обогащается добавлением всех индивидуальных конъюнкций φ_i в качестве признаков. Таким образом, они используют ИЛП для идентификации подмножества конъюнкций из пространства всех возможных конъюнкций исходных признаков. Дополнительным преимуществом использования ИЛП является то, что конъюнктивные предложения h_i также могут быть получены из источников, отличных от тренировочных примеров. Это позволяет легко встраивать знания области в процесс построения признака. После создания исходного набора признаков, они затем используют метод выбора признаков в соответствии с подходом обертки, о котором говорилось выше. На каждой итерации оценивается производительность прогнозирования набора признаков и генерируются два новых набора признаков. Один, исключая худший признак, а другой, добавляя наиболее эффективный признак в другой случайный образец признаков из исходного набора признаков. Полученная модель показала впечатляющий прирост производительности в задаче разрешения лексической многозначности.

2.2.3 Аннотации

Подходы, основанные на аннотациях, позволяют пользователям предоставлять знания области в виде аннотаций вместе с примерами обучения. Затем на основе этих аннотаций будет изучено пространство признаков. Таким образом, исключается необходимость в определении операторов.

Ротт и Смолл [18] предложили интерактивный протокол построения пространства признаков. Вместо того, чтобы предопределять большое пространство признаков с помощью операторов, их подход позволяет создавать динамическое пространство признаков, основанное на взаимодействии между обучающей машиной и экспертом предметной области. Во время учебного процесса, когда учащийся представляет какой-то

пример эксперту по предметной области, эксперт использует знания области, чтобы предоставить дополнительные аннотации (а не только ярлыки). Учащийся теперь должен включить дополнительные знания через модификации пространства признаков и изучить модель. Таким образом, эксперт области может напрямую кодировать знания об области без необходимости определения операторов. Они применили свой метод к проблеме распознавания именованного объекта при использовании семантически связанных списков слов (CCCC) [2, 16] в качестве внешнего знания. CCCC группируют вместе лексические элементы, принадлежащие к одной семантической категории. Например, Направление_Компаса = {восток, запад, север, ... и т.д.}. Протокол работает следующим образом:

1) Учащийся начинает с изучения первоначальной гипотезы h в исходном пространстве признаков F_0 и оценки всех экземпляров, использующих его.

2) Затем функция запроса Q используется для выбора лучших экземпляров и представления их эксперту области. Цель состоит в том, чтобы выбрать Q таким образом, чтобы он минимизировал взаимодействие с пользователем при максимальном воздействии.

3) Эксперт рассматривает экземпляры и отмечает некоторые признаки для абстракции. Например, на рисунке 2 ученик ошибается при маркировке «Chicagoland» как организации.

4) Эксперт отмечает признаки, относящиеся к «западу» для абстракции.

5) Основываясь на этом взаимодействии, пространство признаков изменяется путем замены признаков φ_{east}, φ_{west}, φ_{north} в исходном пространстве признаков F_0 на $\varphi_{Compass}$-$Direction$= $\varphi_{east} \vee \varphi_{west} \vee \varphi_{north}$

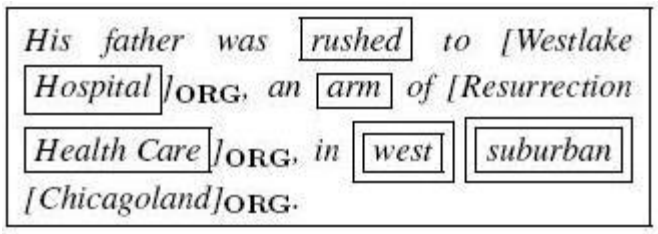

Рисунок 2 – Пример работы протокола CCCC

Рисунок 2 демонстрирует работу протокола. Все лексические элементы с членством в CCCC обведены квадратом. Элементы, используемые для неверного предсказания для «Chicagoland», имеют двойное обведение. Эксперт может выбрать любой обведенный элемент для проверки CCCC.

Другой интерактивный метод принадлежит Рагхавану и Аллену [17], которые представляют тандемный алгоритм обучения для выбора признаков для классификации текста. Алгоритм начинается с небольшого количества помеченных экземпляров и на каждой итерации рекомендует экземпляры и признаки для людей для маркировки. Однако в отличие от CCCC, в их случае пространство признаков остается статичным.

Другие методы, которые позволяют эксперту области напрямую указывать информацию об пространстве признаков с помощью аннотаций, включают [1, 6, 12, 23, 24]. В частности, Лим и др. [12] предлагают метод построения признаков, основанный на объяснительном обучении, для задачи распознавания китайских символов. В их случае знания кодировались в виде «штрихов», которые использовались для генерации символов (или экземпляров). Затем признаки генерировались на основе наличия / отсутствия одинаковых штрихов в разных классах.

2.3 Классификация

Классификация является одной из наиболее изученных проблем машинного обучения и интеллектуального анализа данных [5, 21]. Она состоит в прогнозировании значения категориального атрибута (класса) на основе значений других атрибутов (предсказывающих элементов). Алгоритм поиска используется для индукции классификатора из набора правильно классифицированных экземпляров данных, которые называются обучающей выборкой. Другой набор правильно классифицированных экземпляров данных, известных как множество испытаний, используется для измерения качества полученного классификатора. Различные виды моделей, такие как деревья решений или правила, могут быть использованы для представления классификаторов.

Классификация объекта – номер или наименование класса, выдаваемый алгоритмом классификации в результате его применения к данному конкретному объекту.

В машинном обучении задача классификации относится к разделу обучения с учителем. В контролируемом обучении атрибуты экземпляров данных разделяются на два типа: входы или независимые переменные и выходы или зависимые переменные. Цель процесса обучения состоит в предсказании значения выходов из значения входов. Для того, чтобы достичь этой цели, обучающий набор данных (в том числе экземпляры данных значений входных и выходных переменных с известными значениями) используется для управления процессом обучения. Регрессия и классификация – это два типа контролируемых задач обучения. В регрессии предсказываются непрерывные выходные значения, в то время как при классификации выходы дискретны. В неконтролируемом обучении не существует никакого различия по типу между переменными экземпляров данных. Как следствие, мы не можем говорить об обучающих данных, так как мы не можем иметь набор данных с известным выходом. Целью неконтролируемого обучения является

нахождение внутренней структуры, отношений, или родства, присутствующих в данных. Примерами неконтролируемых задач обучения являются кластеризация и обнаружение ассоциации. Цель кластеризации состоит в том, чтобы разделить данные на различные группы, находя группы данных, которые сильно отличаются друг от друга, либо члены, которых очень похожи друг на друга. Цель ассоциации состоит в нахождении значений данных, которые часто появляются вместе.

2.3.1 Формальная постановка задачи классификации

Пусть X – множество описаний объектов (признаков), Y – конечное множество номеров (имён, меток) классов. Существует неизвестная целевая зависимость – отображение $y^*: X \to Y$, значения которой известны только на объектах конечной обучающей выборки $X^m = \{(x_1, y_1), \dots, (x_m, y_m)$. Требуется построить алгоритм $a: X \to Y$, способный классифицировать произвольный объект $x \in X$.

Однако более общей считается вероятностная постановка задачи. Предполагается, что множество пар «объект, класс» $X \times Y$ является вероятностным пространством с неизвестной вероятностной мерой P. Имеется конечная обучающая выборка наблюдений $X^m = \{(x_1, y_1), \dots, (x_m, y_m)$, созданная согласно вероятностной мере P. Требуется построить алгоритм $a: X \to Y$, способный классифицировать произвольный объект $x \in X$.

3 Генетическое программирование

Многие, казалось бы, разные проблемы в машинном обучении можно рассматривать как требующие компьютерной программы, вычисляющей некоторый требуемый результат в зависимости от входных параметров.

Процесс решения этих проблем можно сформулировать как поиск наиболее подходящей индивидуальной компьютерной программы среди всех возможных компьютерных программ. Пространство поиска состоит из всех возможных компьютерных программ, составленных из функций и терминальных символов, соответствующих проблемной области. Генетическое программирование предоставляет способ поиска этих наиболее подходящих индивидуальных компьютерных программ.

Генетическое программирование пробует решить проблему представления в генетических алгоритмах путем увеличения сложности адаптируемых структур. В частности, адаптируемые структуры в генетическом программировании являются общими иерархическими компьютерными программами, динамически изменяющими размер и вид.

В генетическом программировании популяции сотен или тысяч компьютерных программ генетически выведены. Эта селекция осуществляется с помощью принципа выживания сильнейших и воспроизводства наиболее приспособленных особей вместе с генетической рекомбинацией (скрещиванием) организмов путем применения операций, подходящих для компьютерных программ. Компьютерная программа, которая решает (или приблизительно решает) определенную проблему может возникнуть из комбинации естественного отбора Дарвина и генетических операций.

Генетическое программирование начинается с генерации случайным образом выбранной начальной популяции компьютерных программ из функций и терминалов, соответствующих проблемной области. Функции могут быть стандартными арифметическими операциями, операциями

программирования, математическими, логическими или предметно-ориентированными фикциями. В зависимости от конкретной задачи, компьютерная программа может работать с логическими значениями, целыми, вещественными или комплексными числами, векторами, символами. Создание этой начальной популяции в действительности «слепой» случайный поиск в пространстве проблемной области.

Каждая программа в популяции оценивается, насколько хорошо она выполняет свои задачи в проблемной среде. Эта оценка носит название меры пригодности. Ее вид зависит от проблемы.

Для многих задач пригодность естественно измерять ошибкой, погрешностью компьютерной программы. Чем ближе эта ошибка к нулю, тем лучше данная программа. Также приспособленность может быть комбинацией таких факторов, как корректность, экономность и бережливость.

Как правило, каждая компьютерная программа популяции отработает для нескольких значений входных параметров. Тогда пригодность будет считаться в виде суммы или среднего арифметического значений приспособленности всех входных параметров. Например, пригодность компьютерной программы может быть суммой абсолютной величины от разности вычисленного программой значения и корректного решения проблемы. Эта сумма может быть получена из выборки 50 различных входных значений программы.

За исключением случаев, когда проблема мала и проста, она не может быть легко решена путем «слепого» случайного поиска, компьютерные программы нулевого поколения будут иметь очень плохую пригодность. Тем не менее, некоторые особи в популяции будут немного пригоднее остальных. Эти различия в эффективности следует использовать в дальнейшем.

Принципы репродукции и выживания наиболее приспособленных особей и генетические операции половой рекомбинации (скрещивание) используются для создания нового поколения индивидуальных компьютерных программ из текущей популяции.

Операция репродукции включает в себя селекцию, пропорциональную значениям приспособленности, компьютерных программ из текущей популяции и позволяет отобранным особям выжить путем копирования в новую популяцию.

Генетический процесс полового скрещивания двух родителей – компьютерных программ – используется для создания новых потомков от родителей, выбранных пропорционально значениям пригодности. Программы-родители обычно имеют различный размер и форму. Программы-потомки составляются из подвыражений (поддеревьев, подпрограмм) родителей. Эти потомки, как правило, различаются размером и видом от своих родителей.

Интуитивно, если две компьютерные программы несколько эффективны в решении проблемы, то их части, возможно, тоже немного пригодны. Скрещивая случайно выбранные части относительно пригодных программ, мы можем получить новую компьютерную программу, которая даже лучше решает проблему.

После выполнения операций репродукции и скрещивания с текущей популяцией, популяция потомков (новое поколение) помещается в старую популяцию (прошлое поколение).

Каждая особь новой популяции компьютерных программ затем проверяется на пригодность, и процесс повторяется в течение многих поколений.

На каждом этапе этого параллельного, локально управляемого, децентрализованного процесса состоянием процесса будет являться только текущая популяция особей. Движущая сила этого процесса состоит только в наблюдении за пригодностью особей текущей популяции.

Данный алгоритм будет производить популяцию компьютерных программ, которые через много поколений, как правило, демонстрируют увеличение средней пригодности. Кроме того, эти популяции могут быстро и эффективно приспосабливаться к изменениям в окружающей среде.

Как правило, лучшая особь, которая появляется в любом по счету поколении, обозначается как результат генетического программирования.

Важной особенностью генетического программирования является иерархический характер производимых программ. Результаты генетического программирования по своей природе иерархичны.

Динамическая изменчивость также является важным признаком компьютерных программ, созданных генетическим программированием. Было бы трудно и неестественно пытаться заранее уточнить или ограничить размер и форму возможного решения. Более того, такая предварительная спецификация сужает пространство поиска решений и может исключить нахождение решения вообще.

Еще одной важной особенностью генетического программирования выступает отсутствие или сравнительно малая роль предварительной обработки входных данных и постобработки выходных значений. Входные параметры, промежуточные результаты и выходные значения обычно выражаются непосредственно в терминах естественной терминологии предметной области. Элементы функционального множества также естественны для проблемной области.

И наконец, структуры, подвергающиеся адаптации, активны. Они не являются пассивными кодировками решения проблемы. Вместо этого с учетом компьютера, на котором происходит запуск, программы в генетическом программировании – это активные структуры, способные выполниться в их текущем виде.

Парадигма генетического программирования является независимой от проблемной области. Это обеспечивает единый, унифицированный подход к проблеме нахождения решения в виде компьютерной программы.

ЗАКЛЮЧЕНИЕ

Было проведено исследование решения задачи построения признаков для классификации медицинских изображений методом генетического программирования.

Изучение медицинских изображений позволило предположить, что классификация такого рода данных с помощью интуитивного выделения признаков и применения классификатора будет выдавать результаты с большой погрешностью.

Поэтому были рассмотрены методы, позволяющие автоматически конструировать признаки для классификатора с целью уменьшения погрешности разделения медицинских изображений на классы.

Для решения задачи построения признаков для классификации медицинских изображений был выбран метод генетического программирования. Данный метод был подробно изучен, благодаря чему сформировалось понимание о структуре представления данных в программе, а также операций, применение которых способно привести к решению проблемы.

Далее была разработан алгоритм решения задачи построения признаков на основе генетического программирования. Подробно рассмотрены генетические операторы, их влияние на промежуточный и конечный результаты работы, и на основании проделанной работы выбраны наиболее подходящие операторы репродукции, скрещивания и мутации.

По разработанному алгоритму была реализована программа. Проверено на практике выполнение и результат всех этапов алгоритма.

СПИСОК ЛИТЕРАТУРЫ

1. Druck G, Mann G. S., and McCallum A. Learning from labeled features using generalized expectation criteria. In Sung H. Myaeng, Douglas W. Oard, Fabrizio Sebastiani, Tat S. Chua, Mun K. Leong, Sung H. Myaeng, Douglas W. Oard, Fabrizio Sebastiani, Tat S. Chua, and Mun K. Leong, editors, SIGIR, pages 595–602. ACM, 2008.
2. Fellbaum C. WordNet: An Electronic Lexical Database. MIT Press, 1998.
3. Forman G. An extensive empirical study of feature selection metrics for text classification. J. Mach. Learn. Res., 3:1289–1305, 2003.
4. Guyon I. and Elisseeff A. An introduction to variable and feature selection. J. Mach. Learn. Res., 3:1157–1182, 2003.
5. Han J. and Kamber M., Data Mining – Concepts and Technique (The Morgan Kaufmann Series in Data Management Systems), 2nd ed. San Mateo, CA: Morgan Kaufmann, 2006.
6. Huang Y. and Mitchell T. M. Text clustering with extended user feedback. In Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 413–420. SIGIR, 2006.
7. Kohavi R. and John G. H. Wrappers for feature subset selection. Artif. Intell., 97(1-2):273–324, 1997.
8. Koza J. R. Genetic programming: on the programming of computers by means of natural selection. MIT Press, Cambridge, MA, USA, 1992.
9. Kramer S., Lavrac N. and Flach P. Propositionalization approaches to relational data mining, pages 262–286, 2000.
10. Langdon W. B. Genetic programming and data structures. Department of Computer Science, University College London, 1996.
11. Lavrac N., Dzeroski S. and Grobelnik M. Learning nonrecursive definitions of relations with linus. In EWSL-91: Proceedings of the European

working session on learning on Machine learning, pages 265–281, New York, NY, USA, 1991. Springer-Verlag New York, Inc.

12. Lim S. H., Wang L. L. and DeJong G. Explanation-based feature construction. In IJCAI07, the Twentieth International Joint Conference on Artificial Intelligence, pages 931–936, 2007.

13. Markovitch S. and Rosenstein D. Feature generation using general constructor functions. Mach. Learn., 49(1):59–98, 2002.

14. Matheus C. and Rendell L. A. Constructive induction on decision trees. In Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, pages 645–650. Morgan Kaufmann, 1989.

15. Pagallo G. Learning dnf by decision trees. In Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, pages 639–644. Morgan Kaufmann, 1989.

16. Pantel P. and Lin D. Discovering word senses from text. In Proceedings of ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pages 613–619, 2002.

17. Raghavan H. and Allan J. An interactive algorithm for asking and incorporating feature feedback into support vector machines. In SIGIR'07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval, pages 79–86, New York, NY, USA, 2007. ACM.

18. Roth D. and Small K. Interactive feature space construction using semantic information. In Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009), pages 66–74, Boulder, Colorado, June 2009. Association for Computational Linguistics.

19. Specia L., Srinivasan A., Joshi S., Ramakrishnan G., and Nunes M. G. V. An investigation into feature construction to assist word sense disambiguation. Mach. Learn., 76(1):109–136, 2009.

20. Specia L., Srinivasan A., Ramakrishnan G., and Nunes M. D. V. Word sense disambiguation using inductive logic programming, pages 409–423, 2007.

21. Tan P.-N., Steinbach M., and Kumar V., Introduction to Data Mining. Reading, MA: Addison-Wesley, 2005.

22. Yang D., Rendell L., and Blix G. A scheme for feature construction and a comparison of empirical methods. In Proceedings of the Twelfth International Joint Conference on Artificial Intelligence, pages 699–704. Morgan Kaufmann, 1991.

23. Zaidan O. F. and Eisner J. Modeling annotators: A generative approach to learning from annotator rationales. In EMNLP, pages 31–40. ACL, 2008.

24. Zaidan O. F. and Eisner J. Using annotator rationales to improve machine learning for text categorization. In In NAACL-HLT, pages 260–267, 2007.