Министерство образования Российской Федерации
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Факультет информатики
Кафедра прикладной информатики

ДОПУСТИТЬ К ЗАЩИТЕ В ГАК
Зав. кафедрой прикладной информатики,
доктор технических наук, профессор
_____ С.П. Сущенко
«___» июня 2002 г.

Шевелев Олег Геннадьевич

Установление авторства текста методами искусственного интеллекта

Дипломная работа

Научный руководитель:
канд. техн. наук
_____ В.В. Тютерев

Автор работы
_____ О.Г. Шевелев

Томск 2002

РЕФЕРАТ

Дипломная работа, 57 стр., 8 рис., библ. 23 назв.

ОПРЕДЕЛЕНИЕ АВТОРСТВА ТЕКСТОВ, КОМПЬЮТЕРНАЯ ОБРАБОТКА ТЕКСТОВ, СТИЛЕМЕТРИЯ, АВТОРСКИЙ ИНВАРИАНТ, ПЛАГИАТЫ, НЕЙРОННЫЕ СЕТИ, ГЕНЕТИЧЕСКИЕ АЛГОРИТМЫ, КЛАСТЕРИЗАЦИЯ ТЕКСТОВ.

(1) Объекты исследования – текстовые данные, авторский инвариант.

(2) Цели работы – реализация методов установления авторства текстов с фиксированным набором альтернатив и кластеризации текстов.

(3) Методы исследования – теоретический и экспериментальный (на ЭВМ).

(4) Основные результаты – реализован метод установления авторства текстов с фиксированным набором альтернатив на базе нейронных сетей прямого распространения; разработан и реализован метод кластеризации текстов, основанный на генетическом поиске набора характеристик.

Оглавление

ВВЕДЕНИЕ

С увеличением текстовых массивов данных открываются новые пути в исследовании литературы и языка. Статистические методики с поддержкой компьютерных технологий обладают огромным потенциалом в разрешении многих теоретических задач лингвистики и практических задач обработки текстовых данных. Одним из направлений компьютерного анализа текстов является стилеметрия[1], занимающаяся оценкой стиля во всех его возможных интерпретациях. Задача автоматического установления авторства (authorship attribution) принадлежит разделу стилеметрии, выявляющему особенности проявления авторского стиля. Первые попытки ее разрешения относятся к середине XIX века. Методики, инструменты и акценты в исследованиях претерпели существенные изменения, но суть проблемы остается прежней: по каким-либо статистическим характеристикам (морфемным, стилистическим, семантическим и др.) необходимо установить, кто является автором того или иного текста. Допускаются расширения задачи, в которых производится кластеризация текстов исходя из авторской принадлежности, таким образом что тексты каждого автора образуют выделенную группу (кластер).

Настоящая работа является исследованием как в области нахождения характеристик, являющихся уникальными для каждого конкретного автора, так и методов их формирования и интерпретации результатов. В качестве базового инструмента исследования и обработки данных выбран аппарат нейронных сетей, обладающий гибкостью в настройке и простотой в применении. Помимо теоретических выкладок и рассуждений, содержащих подробное рассмотрение поставленной проблемы и обзора имеющихся методов, в дипломную работу включено описание реализации программы автоматического установления авторства, при наличии заданных альтернатив.

[1] В отечественной литературе встречается также «стилостатистика», англоязычный вариант “Stylometry”

1. ПРОБЛЕМА ОПРЕДЕЛЕНИЯ АВТОРСТВА ТЕКСТА

1.1. Суть проблемы и предпосылки ее разрешения

Существует различные подходы к постановке задачи автоматического установления авторства. Можно связать ее с более широким вопросом об атрибуции заданного текста [1]. В данном случае говорят о присвоении тексту всевозможных атрибутов, включающих не только имя его создателя, но и время или период написания, жанр, и т.п. В ходе исследования систематизируются минимальные характеристики текста и приводятся к виду описательных статистик, дающих представление об искомых атрибутах. На основе собранных данных ставятся различные гипотезы и даются предсказания о характере тех или иных индивидуальных особенностях текста.

Еще один из возможных подходов вовлекает понятие стилеметрии. За основу берутся все те же описательные статистики, но результаты получаются не в качественной шкале, как в случае атрибуции текстов (например, похож/непохож, Пушкин/Лермонтов, XIIIX/XIX век), а в количественной. Таким образом, предпринимается попытка измерить стилевые особенности текста, в частности стиль автора. Методы стилеметрии имеют поддержку в глазах лингвистов. Многие специалисты в языкознании давным-давно подошли к мысли количественного описания стилей вместо традиционного размытого и неточного качественного [2]. Делая оценки авторского стиля в количественной шкале, можно группировать результаты, вычислять близость авторских стилей, говорить об устойчивости авторского стиля на ряде произведений, т.е. иметь возможность более широкой интерпретации.

Основным предположением, без которого невозможно какое-либо автоматическое установление авторства, является то, что любой текст обладает некоторым набором скрытых свойств, уникальных для каждого зрелого автора. Эти свойства вносятся неосознанно, и не зависят от желания писателя, поэтому практически не поддаются подражанию.

Сложность задачи заключается в нахождении таких характеристик и правильной их интерпретации. Стилеметристы нередко делают ошибки, пытаясь обобщить частные результаты измерений на более широкие случаи [3]. Избежать этого можно лишь тестированием свойств найденных статистик на больших объемах данных [4]. Отсюда вытекает первое требование к эффективности методик анализа особенностей текста и наличии необходимого количества текстов для проверки.

Второе требование заключается в полноте статистического описания. Очевидно, что одной или двух характеристик для универсальной методики оценки авторского стиля недостаточно (хорошие результаты на одной выборке не обязательно дадут тот же результат

на другой), нужно уметь извлекать информацию из целого набора описательных статистик, делать выводы исходя из взаимосвязей между ними. В данном случае оказываются полезными методы искусственного интеллекта, позволяющие специальным образом «обучать», «тренировать» программу принятия решений или делать самостоятельные выводы на основе классификации данных, редукции их размерности [5].

1.2. Возможные области применения

1.2.1. Анализ текстов

Наиболее очевидным применением методик определения авторского стиля является возможность статистического описания текстов. Для неспециалистов данный инструмент ничего не дает, но лингвистам, несомненно, будет интересно провести исследования касающиеся феномена авторства: в чем состоит отличие в стиле того или иного писателя, что делает динамичной/увлекательной/легко читаемой его речь, какие характеристики являются индивидуальными, а что можно считать общностью. Это лишь небольшой спектр вероятных вопросов. Таким образом, решение задачи установления авторского стиля представляет собой теоретическую ценность.

Практические задачи исследования можно выделить в целевые группы.

Первая группа относится к судебной практике. Даже непосвященным известен нашумевший процесс с «Тихим Доном» [6], многие задумывались над творениями гения Шекспира. Помимо вопросов о плагиате и авторском праве интерес могут представлять письменные показания свидетелей, письма самоубийц, утверждения сделанные под давлением [7]. Существуют методики, которые уже используются в судебной практике, но многие независимые исследователи говорят об их несостоятельности и ненадежности.

Вторая выделенная нами группа – исследование литературы и история. Существует целый ряд анонимных произведений с неустановленным или спорным авторством. Методики определения авторства призваны разрешить эти неоднозначности. С исторической точки зрения важно уметь связывать различные архивные документы с автором и периодом их написания. Только в этом случае можно делать какие-либо выводы на основе содержания исторических текстов.

Третья группа – сфера образования и психология. Ни для кого не секрет, что с развитием мировой сети школьники и студенты все меньше работают самостоятельно, обращаясь к уже готовым рефератам, курсовым или докладам. Цитированные куски текста могут превышать вклад самого автора и зачастую не содержат указания на источник. Методами определения авторства можно выявить подобные «учебные плагиаты», тем самым осуществляя контроль и выставляя аргументированную оценку за работу. Помимо этого

возможны исследования возрастных категорий учащихся, развития их речи с годами обучения, изменения в стиле письма. Грамотный психологический и лингвистический подход не исключает возможности ответа на такие вопросы, как «что считать хорошим/качественным стилем письма?», «в чем заключаются огрехи стиля того или иного ученика?», «в чем особенности его речи?». Ответы на данные вопросы сделают значительный вклад в объективное оценивание сочинений, которое в настоящее время в значительной мере подвержено субъективным и далеко не всегда справедливым взглядам проверяющих.

1.2.2. Синтез литературного стиля

Обратная и более сложная задача подразумевает компоновку авторского стиля на основе полученных данных. Она может пригодиться при коллективном переводе или писательском соавторстве, когда необходимо сгладить различия между участниками проекта и привести текст к однородному виду. При наличии информации о свойствах, характеризующих авторский стиль, не исключена возможность написания программы-советчика, помогающей приблизить собственный письменный труд к произведениям какого-либо писателя, либо справиться со стилистическими проблемами в письме. Кто знает, может быть в скором времени станет вполне реальным обучиться психологичности языка Грина или красочности Набоковских произведений?

2. ИЗВЕСТНЫЕ ПОДХОДЫ К ЗАДАЧЕ ОПРЕДЕЛЕНИЯ АВТОРСТВА ТЕКСТА

2.1. Субъективно-атрибутивная методика

Данная методика не относится к методам автоматического установления авторства, и решение выносится человеком. Интересна она с точки зрения своей очевидности. К ней прибегают все неспециалисты, так или иначе встречающиеся с проблемой установления авторства. В ходе субъективно-атрибутивной методики происходит фиксация внешних деталей авторского стиля – любимых слов и оборотов автора, тематики его произведений, простоты или сложности речи, а также других видимых характеристик. Выбор признаков остается на совести оценивающего и не может гарантировать ошибки в случае нехарактерного для автора произведения или подражания. К тому же нередки случаи, когда у автора вообще нет четко выраженных внешних признаков, или, например, он сознательно меняет свой стиль и тематику от произведения к произведению.

2.2. Формально-количественный подход

Под названием «формально-количественный подход» можно объединить все методы так или иначе связанные с нахождением и оценкой характеристик, вносимых автором бессознательно, и, соответственно, не поддающихся подражанию или существенному изменению[2]. Точность их оценки возрастает с объемом текста.

2.2.1. Попытки начала века: лингвистические спектры

Метод, впервые сформулированный Н.А.Морозовым в начале XX века в его работе «Лингвистические спектры: средство для отличения плагиатов от истинных произведений того или иного неизвестного автора» [3], привлекает внимание исследователя к проблеме автоматического и формального распознавания авторства, отклоняя субъективные попытки оценки, не подкрепленные математическими аргументами: «Чисто субъективный, основанный лишь на индивидуальной чуткости, способ отгадывания авторов не может иметь серьезного научного значения, так как он не дает безусловных доказательств, обязательных для каждого». В качестве характеристик стиля Н.А.Морозов предлагает брать часто используемые слова – предлоги, союзы, частицы, подсчитывая число употреблений каждой в отдельности. Он выстраивает графическое отображение стиля – лингвистические спектры и пытается сравнивать работы различных авторов. Даже на нескольких характеристиках

[2] Не исключено, что с годами творческой работы и эти характеристики могут изменяться. Тем не менее, их изменения не должны быть настолько велики, чтобы происходила заметная путаница в распознавании.

некоторые авторы поддаются разделению, но есть и такие, которые имеют неустойчивый лингвистический спектр (в исследованиях Морозова это Л.Н.Толстой).

Основным недостатком работы являются мизерные масштабы экспериментов. Надо отдать должное Н.А.Морозову, в его время ни о каких компьютерах не было и речи, и все подсчеты производились вручную. Тем не менее, его выводы не раз подвергались сомнениям (см. ответ академика А.А.Маркова [4]) и, в конечном счете, выяснилось, что лингвистические спектры слишком неустойчивы, чтобы служить серьезным основанием для разграничения авторского стиля. Тем не менее, это не исключает их вспомогательной роли.

К положительным моментам можно отнести интересную идею с выделением приведенных, а не относительных спектров. В ходе пространных рассуждений, Н.А.Морозов приходит к выводу, что подсчитывать спектры как таковые нецелесообразно и разумней соотносить их со спектрами литературной эпохи, выявляя таким образом коэффициенты индивидуальности авторов в употреблении того или иного слова.

Идеи Морозова получили новое воплощение в методе опорных слов Фоменко [5].

2.2.2. Метод опорных слов. Эксперименты Фоменко

Самым существенным, на наш взгляд, в работе Фоменко является грамотная постановка задачи. На протяжении всей работы автор пытается найти характеристику, названную им АВТОРСКИМ ИНВАРИАНТОМ. Критерии характеристики оцениваются двумя ключевыми правилами:

1. Для одного автора данная характеристика должна иметь минимальную вариацию, т.е. ее колебания могут быть признаны несущественными;

2. Для разных авторов отклонение характеристики от своего среднего должно быть существенным, чтобы было возможным статистически верно разделить двух разных авторов;

3. «Она должна быть достаточно "массовой", интегральной, чтобы СЛАБО КОНТРОЛИРОВАТЬСЯ автором на сознательном уровне. Другими словами, она должна быть его "бессознательным параметром", коренящимся настолько глубоко, что автор даже не задумывается о нем. А если бы даже задумался, то не смог бы долго его контролировать и в результате довольно быстро вернулся бы в прежнее устойчивое и ТИПИЧНОЕ для него состояние» [5].

В ходе серьезной и кропотливой работы по исследованию десятков авторов с множеством их произведений (вычисления опять же производились вручную) была найдена очень простая и эффективная характеристика – процент содержания служебных слов в тексте. В нее вошло 55 служебных слов (14 союзов, 38 предлогов и 17 частиц, см. Приложение 1). Характеристика варьируется от 16% до 30% в целом, и в пределах 1-2% для

конкретного автора. Таким образом, она обладает поразительной устойчивостью и в ходе творчества писателя на протяжении всей его жизни практически не изменяется. Различие в 2-3% для двух текстов может быть серьезных основанием полагать, что их писали разные авторы.

В качестве демонстрации своего метода Фоменко пытаются разобраться с авторством «Тихого Дона» и в итоге выдвигают достаточно обоснованное обвинение М.А.Шолохова в плагиате.

Несмотря на серьезные результаты, метод опорных слов не лишен недостатков и, безусловно, не решает всех проблем. Прежде всего, оценка процента служебных слов может даваться лишь на огромных, заведомо однородных выборках, в терминах современной компьютерной техники – это тексты размером свыше 100 килобайт. Далеко не всегда имеется возможность такого масштабного исследования. Еще более серьезным ограничением является чрезвычайно низкая разделительная способность оценки в случае большого числа авторов. Напомним, что она варьируется в среднем от 16 до 30, т.е. потенциально может разделять лишь 14 авторских стилей. Но для получения универсального метода нужно рассчитывать на десятки, сотни авторов. Таким образом, метод опорных слов – лишь вспомогательный инструмент, обладающий достаточным, но не необходимым качеством разделения.

2.2.3. Метод накопительных сумм (QSUM[3])

Метод накопительных сумм изначально не был предназначен для нужд стилеметрии. Он использовался как мера контроля качества в промышленности. С помощью него измерялись отклонения качества от среднего и отслеживались эффекты, которые вносили эти отклонения. Преимущество QSUM заключаются в том, что он дает отклонения хронологически, отображая их накопленную сумму, а не просто одиночное число.

Интерпретация метода в терминах стилеметрии не вызывает трудностей. Вместо того чтобы измерять отклонения качества, нужно отслеживать колебания стилеметрических характеристик, найденных в тексте.

Идея и результаты первых исследований принадлежат А.К. Мортону [8], проводившему свои эксперименты еще в середине XX века. В дальнейшем его метод перепроверялся и подтверждался множеством исследователей [7, 9, 10], находились как положительные, так и отрицательные отзывы. Несмотря на то, что некоторые независимые исследователи указывают на неустойчивые результаты данного метода, он до сих пор используется в судебной практике Великобритании [11].

[3] Также встречается как “cusum”

Основная задача метода накопительных сумм заключается в проверке текст на однородность стиля. Таким образом, QSUM позволяет установить, писался тот или иной текст одним или несколькими людьми. Отсюда наиболее вероятное применение метода выявление плагиатов. В том случае, если автор текста пользовался «дополнительными источниками», результаты QSUM дадут о себе знать. Для того чтобы проверить писались ли два различных текста одним человеком (что равносильно сравнению авторских стилей) достаточно соединить два текста в один и с помощью QSUM измерить однородность. Подобными комбинациями можно значительно расширить область применения метода.

Идея метода заключается в следующем. Выбираются две или более стилеметрических характеристик, являющихся функциями предложения. Далее, производится расчет этих характеристик для каждого предложения, вычисляются средние значения этих характеристик. Считаются отклонения от средних значений для каждого предложения, и строится накопительная сумма этих отклонений (начинаем с нуля, затем прибавляем отклонение первого предложения, за ним второго и т.д.). Строится масштабированный график накопительных сумм для каждой характеристики. Для однородного стиля графики характеристик должны практически совпадать, тогда как неоднородный текст покажет их несовпадение.

Практика Мортона показала, что наиболее удачными характеристиками для текста на английском языке являются следующие две стилеметрические функции:

1. Длина предложения;

2. Число двух и трехбуквенных слов + число слов, начинающихся с гласной буквы (причем, если слово начинается с гласной буквы и состоит из, например, двух букв, то оно учитывается лишь один раз).

Слабость метода заключается в неоднозначности процедуры установления различия в графиках характеристиках. Само по себе масштабирование (без которого невозможно обойтись) уже вносит значительные искажения, кроме того, не совсем ясно, какую степень отклонения считать существенной. Даже если не учитывать недоказанность состоятельности метода, он слишком узок в своем применении и не годится для обработки больших текстовых массивов (получается, тогда нужно будет комбинировать между собой десятки текстов!). А, если верить одному из ведущих специалистов QSUM Джиллу Фарингдону, QSUM дает хорошие результаты максимум на 50 предложениях, исключая возможность обработки произведений в целом.

2.3. Методы искусственного интеллекта

2.3.1. Использование нейронных сетей прямого распространения

Одним из первых опытов по использованию нейронных сетей в стилеметрии является работа [12]. В ней использовались классические персептроны прямого распространения. Идея исследователей заключалась в том, чтобы подавать на входы нейронной сети специально собранные характеристики, производные текста, а на выходе формировать вектор принадлежности тому или иному автору. Обучение осуществлялось с учителем, таким образом, с каждым вектором-входом (статистическими характеристиками текста) ассоциировался вектор-ответ (авторский стиль). В дальнейшем, когда сети подавали статистику неизвестного ей текста, она пыталась делать выводы самостоятельно и «говорила», кто является автором.

В качестве характеристик текста брались одиннадцать функциональных слов английского языка (an, any, can, do, every, from, his, may, on, there, upon), как самые употребительные и независящие от контекста. Исследователи пытались разобраться со знаменитыми Federalist papers, авторство которых неизвестно, и сопоставляли их с работами двух известных политиков Мэдисона и Гамильтона. В итоге сеть сделала однозначные выводы в пользу Мэдисона.

В наших первоначальных исследованиях был применен данный подход с некоторыми модификациями (см. гл. 4.Установление авторства текстов при заданном наборе альтернатив).

2.3.2. Использование RBF-сетей

Лоуи и Мэтьюс [13] использовали RBF-сети в стилеметрическом анализе. Данный тип нейронных сетей имеет ряд преимуществ перед стандартными сетями прямого распространения. RBF-сети легче поддаются интерпретации, для них даже существуют определенные методы извлечения правил. Кроме того, какая-либо предварительная информация может использоваться в качестве отправной точки для их тренировки, что очень важно для маленьких наборов данных, случая достаточно распространенного в задачах автоматического установления авторства.

Лоуи и Мэтьюс использовали пять функциональных слов для тренировки сети (are, in, no, of, the) и тестировали свой метод на пьесах Шекспира и Флетчера. В сумме было получено по 50 примеров для каждого автора. Результаты классификации соответствовали традиционным исследованиям в области литературы.

3. АНАЛИЗ ПРОБЛЕМ В РЕШЕНИИ ЗАДАЧИ УСТАНОВЛЕНИЯ АВТОРСТВА

Данная глава посвящена общим вопросам, с которыми приходится сталкиваться исследователям методов автоматического установления авторства. В ней намечены проблемы, затрагивающиеся в литературе по стилеметрии, и дано наше собственное толкование. К сожалению, время разрешения этих вопросов еще не пришло. Каждый раз, в каждом конкретном случае исследователю приходится полагаться на собственные силы и находить частное, во многих случаях ненадежное, решение.

3.1. Материал для обучения и тестирования

Авторский стиль выражается в творчестве автора, и его загадка кроется в том наборе символов, который называется текстом. Таким образом, чтобы научиться распознавать автора, нужно четко сформулировать, как должны обрабатываться произведения, какой объем достаточен для распознавания, весь ли текст годится для обработки (или, например, следует упускать из виду прямую речь персонажей, отдавая предпочтение речи автора), какое количество распознанных текстов может говорить об эффективности метода. На все эти вопросы нами не было найдено однозначного ответа. Исследователи либо действуют умозрительно, либо методом проб и ошибок достигают какого-то промежуточного, но результативного варианта, оставляя свой выбор без доказательств.

В случае с объемами выборочных данных (размеры блоков текста, которые подвергаются статистической обработке) многие советуют придерживаться принципа «чем больше, тем лучше». С точки зрения статистики все верно – больший объем выборки более полно представляет генеральную совокупность, которая включает все произведения автора, написанные и те, которые еще будут написаны, но с точки зрения информатики в таком подходе есть свои сложности. Даже на текущем высоком уровне технического оснащения объемы данных играют существенную роль. До сих пор нет возможности загрузить все имеющиеся произведения (лежащие, как правило, в электронных библиотеках Интернет или на компакт-дисках) и провести их полномасштабный анализ. Поэтому вопрос о минимальном объеме выборки актуален.

Говоря об объемах текста, необходимо согласовать в каких единицах их измерять. Самое простое решение – килобайты (байты, мегабайты), но кроме него можно предложить измерять текст количеством слов, предложений или абзацев, то есть в величинах имеющих смысловую связь с предметом исследования. Наиболее разумное решение – слова. Килобайты слишком отвлеченное понятие, не отражающее специфики анализа, длина

предложений и абзацев очень часто варьируются, и, кроме того, большая часть описательных характеристик текста связано именно с понятием слова (длины слов, части речи, части предложения, служебные слова и т.д.).

В работе Фоменко [5] обрабатываются выдержки в 16000 слов, что составляет приблизительно 105-120 Кб электронной версии или 60-70 страниц печатного текста. Далеко не каждое литературное произведение имеет такой объем, и при таком подходе огромные массивы рассказов и небольших повестей ускользают от анализа. В данной работе по умолчанию использовались выборки в 6000 слов (40-50 Кб, 20-30 страниц), что позволяет исследовать значительную часть малой прозы. У нас, как и у прочих других, нет доказательств эффективности подобных разбиений. Данная цифра была выбрана только из соображений практичности. Она относительно мала, чтобы существенно не ограничивать себя в исследованиях, и достаточно велика для существования возможности разграничения авторов.

В ходе поиска авторского инварианта может понадобиться производить обучение на примерах. То есть, заранее зная, кто автор разбираемого текста, пытаться делать выводы о надежности той или иной характеристики. В данном случае также необходимо знать какое количество выборок будет достаточным для объявления характеристики устойчивой для конкретного автора, и сколько авторов нужно проверить, чтобы объявить характеристику существенно варьирующейся для разных писателей. Теме нахождения оптимальных параметров подобного поиска можно посвятить отдельную работу, настолько она сложна и мало исследована. Наши эксперименты были ориентированы на вычислительные мощности имеющихся компьютеров и проводились с как можно большим количеством примеров. Тем не менее, раз вопрос о размерах выборки и количестве примеров для обучения остался без доказательства, полученные результаты нельзя обобщить на всех возможных писателей.

Не лучшая ситуация и относительно избирательности в ходе накопления статистики. Каждый автор накладывает свои собственные ограничения на текст и изобретает правила обработки. Например, в QSUM собственные имена и названия считаются всегда одним словом, даже если они многосоставные («Собор Парижской Богоматери», «Владимир Владимирович Путин»), в экспериментах Фоменко прямая речь героев не подвергается статистической обработке. Если редукция имен отчасти обоснована (в большинстве случаев автор не имеет свободы в употреблении имен, они есть такие, какие они есть, к ним очень редко можно быть подобраны синонимы или переформулировать их иначе), то избавление от прямой речи – вопрос спорный. Исключение прямой речи накладывает серьезные ограничения на объемы выборки (если брать только слова автора, то произведение становится еще меньше) и усложняется алгоритм разбора. Кроме того, нет никакой гарантии

потери информации о писателе в данном случае. Речь героев независима от слов авторов лишь в самом идеальном случае, на практике же она наверняка содержит в себе информацию о создателе. Таким образом, в данной работе было решено не искажать произведения и анализировать исходный текст.

3.2. Проблема выбора набора характеристик и их оценки

Фактически, задачу установления авторства можно свести к поиску статистик[4], обладающих свойствами авторского инварианта. Когда нужный авторский инвариант (тот самый набор характеристик) найден, определить, кто является автором текста, гораздо проще. А значит, раз до сих пор нет единого подхода к установлению авторства, то универсальный авторский инвариант не найден. Остается задать вопрос, а существует ли он вообще? Но нет доказательства и на этот счет. На практике же в основном решается ограниченный круг задач [8, 12, 13], связанный с определением авторского инварианта для предварительно заданного набора текстов. Настройка, тестирование и демонстрация инструментов анализа ориентирована лишь на эти тексты, и нет никакой гарантии, что методы будут эффективно справляться с задачей на других данных. Например, авторский инвариант Фоменко [5] (количество служебных слов в тексте) также не обладает универсальной разделительной способностью, но вполне справляется с ограниченным числом текстов, удовлетворяющим условиям объема (рекомендуемый минимум – 16000 слов).

Решив определять авторство с помощью статистической оценки, нужно среди бесчисленного количества характеристик выбрать те, которые будут использованы при сборе статистики и анализе текстов. Можно составлять какие угодно сложные характеристики, считать хоть букву «Ю», на которую заканчивается слово, стоящее через одно слово от слова с буквой «Я» на первом месте, но основная часть исследователей ограничивается их стандартным набором. Он содержит длины слов, длины предложений, служебные слова, частоты встречаемости отдельных букв, буквосочетаний, знаков препинания, закономерности в повторении этих единиц, сложность и частоты появления отдельных грамматических конструкций. По результатам первой части нашей практической работы можно убедиться, что для разделения фиксированного множества авторов достаточно стандартного набора характеристик и их комбинаций. Тем не менее очевидно, что для построения универсального и независящего от текстов авторского инварианта необходимо искать новые пути формирования характеристик. Во второй части работы предлагается алгоритм, реализующий эту возможность.

[4] Статистика – любая функция выборочных данных

Говоря о сборе информации на основе статистик, нельзя не упомянуть, что существуют также методы, базирующиеся на других принципах оценки. Среди них контент-анализ, в ходе которого происходит как бы осмысление текста компьютером, а потом предъявление семантической оценки, а также метод, работающий со словарным запасом автора. Несмотря на их кажущееся отличие, по существу это лишь макроуровни метода сбора статистик. Текст в любом случае состоит из букв, пробелов и знаков препинания, и в этом плане можно даже провести аналогию с машиной Тьюринга. Как к передвижению ленты и пересылке информации в модели машины Тьюринга можно свести любой алгоритм, так и контент-анализ с методом словарей сводится к методу сбора статистик. Все слова авторского словаря можно превратить в характеристики, тогда для одного автора они будут одни, для другого они будут отличны от первых. Семантика определяется строением и расположением слов, а значит, существуют характеристики, позволяющие сделать замеры «смыслового слоя текста». Вопрос состоит лишь в том, что, если воспользоваться тем же механизмом, что определяет длины слов и расположения букв, то получатся слишком громоздкие описания. Поэтому, несмотря на сводимость, некоторые методы уместно рассматривать отдельно.

Установив набор характеристик, исследователь сталкивается с проблемой их структуризации, в чем существенную помощь может оказать классические статистические методы. С помощью факторного анализа и анализа главных компонент можно установить вклад той или иной характеристики в процесс распознавания автора, иерархический кластерный анализ позволит сделать объединение отдельных характеристик в подгруппы, подгрупп в группы и так далее. Немалую помощь можно получить от нейронных сетей прямого распространения, если попытаться обучить сеть на наборе примеров, взяв в качестве входов отдельные характеристики, а затем оценивать, какое влияние оказывает тот или иной вход на систему выходов.

Разобравшись с влиянием характеристик на процесс распознавания, нужно исключить ненужные и повторяющие друг друга характеристики, подсчитав корреляции между ними. Возможно, в ходе оценки предстоит проделать реструктуризацию характеристик (некоторые из них, например, создадут агрегированные показатели) или провести преобразование в случае вращения факторного пространства для лучшего представления накопленных результатов. Таким образом, отбор и приведение характеристик в порядок – сложный и трудоемкий процесс начального этапа, который в идеале необходимо автоматизировать. За время работы с проблемой установления авторства нами был найден алгоритм, позволяющий осуществлять целенаправленный перебор множества характеристик, и проблема автоматизации их отбора была частично решена (см. гл. 5.Кластеризация текстов).

4. УСТАНОВЛЕНИЕ АВТОРСТВА ТЕКСТОВ ПРИ ЗАДАННОМ НАБОРЕ АЛЬТЕРНАТИВ

4.1. Постановка задачи

Бывает, что необходимо решить конкретный вопрос, и нет смысла искать универсальные методы, которые будут работать на любых данных. Например, не так давно перед литературной общественностью стояла задача узнать действительно ли «Тихий Дон» произведение М.А.Шолохова, либо писатель осуществил плагиат. Примерно такая же ситуация обстояла с Federalists paper [12] и произведениями Шекспира [13]. Отсюда видно, что в основном это проблемы спорного авторства.

В подобных случаях успешно работают так называемые методы атрибуции при заданном (фиксированном) наборе альтернатив [12, 13]. Выбирается то множество авторов, произведения которых необходимо классифицировать, и выбираются тексты из числа работ выбранных писателей, для которых автор доподлинно известен. Происходит фиксация особенностей каждого писателя на основе заданного набора статистик, в результате чего становится возможным делать выводы, чьему перу из выбранного множества писателей принадлежит тот или иной текст со спорным авторством.

4.2. Предварительно заданный авторский инвариант и тренируемый метод подсчета

Первый шаг к разрешению проблемы в ходе данной работы был сделан именно в области установления авторства с заданным набором альтернатив, так как это более простая задача по сравнению с поиском универсального набора характеристик, позволяющий идентифицировать любого автора. Метод, заложенный в программе, схож с идеей работы [12]. Была также использована нейронная сеть прямого распространения и произведено ее обучение с учителем. Изначально подразумевалось, что сеть может научиться любым знаниям по примерам и способна выбрать из всего потока входов действительно значимые компоненты. В отличие от реализаций работы [12], где были взяты лишь функциональные слова, нами был выбран путь максимального наращивания входов, в число которых были включены всевозможные характеристики. Были взяты следующие статистические характеристики текста:

1. Доли появления отдельных служебных слов (55 служебных слов, предложенных Фоменко, среди них 14 союзов, 24 предлога, 17 частиц; см. Приложение 1);

2. Доли появления внутренних знаков предложения (двоеточие, запятая, тире, точка с запятой, скобки, кавычки);

3. Доли появления внешних знаков предложения (точка, восклицательный знак, вопросительный знак, многоточие, комбинации восклицательных и вопросительных знаков с многоточием);

4. Доли предложений определенной длины (доля длин предложений из одного слова, из двух слов, и так далее до 30);

5. Доли слов определенной длины (доля длин слов из одной буквы, из двух букв, и так далее до 10 букв);

6. Доля появления союзов;

7. Доля появления предлогов;

8. Доля появления частиц;

9. Доля всех служебных слов в тексте;

10. Средняя длина фразы[5].

Итого, в первоначальном варианте было 111 входов. В процессе дальнейших исследований два входа пришлось удалить (оказалось, что их дисперсия равна нулю), переделать логику подсчета некоторых долей, а также расширить понятие средней длины фразы до 20 параметров, которые обозначают доли фраз определенной длины в тексте. Конечный вариант включал в себя 132 параметра.

Таким образом, данный метод использует предварительно заданный набор характеристик, что существенно ограничивает его надежность. Более того, качество метода определяется только тем, насколько он в конечном итоге сможет отделить одного писателя от другого. Может оказаться и так, что обученная на примерах нейронная сеть (а примерами являются фактически не сами тексты, а накопленная по ним статистика, заданная выбранным множеством характеристик) не справится с поставленной задачей. Набор характеристик – узкое место метода, от которого напрямую зависит его эффективность. Проверка разделительной способности выбранного набора характеристик – очень трудоемкая и сложная процедура, без прохождения которой, безусловно, нельзя строить серьезных выводов. Так как данный метод не является ключевым в данной работе, оцениванию характеристик было посвящено лишь несколько пробных экспериментов с помощью пакета Statistica. Была собрана статистика на основе выбранных характеристик и проведен факторный и кластерный иерархический анализы, что помогло найти и исправить ошибки в подсчетах, сделать выводы о дублировании информации в некоторых характеристиках, а

[5] Среднее число слов между двумя любыми знаками препинания

также отработать механизм анализа характеристик с помощью мощного статистического пакета.

Конечным результатом работы алгоритма по определению авторства текста с фиксированным набором альтернатив выступает вектор длиной равной числу писателей и значениями, позволяющими установить, кто автор спорного текста. В отличие от предварительно заданного набора характеристик, метод подсчета конечного результата генерируется автоматически. Он формируется в процессе обучения нейронной сети и является, по сути, множеством весов в узлах решетки нейронов. Программе установления авторства требуется лишь подать на входы обученной нейронной сети вектор результатов статистической обработки, собранной на основе выбранных характеристик, и «пропустить» через нейронную сеть (на практике это реализуется несколькими матричными операциями), а на выходе получить вектор результатов, позволяющих установить авторство.

4.3. Этапы разрешения проблемы (руководство пользователя)

Данный раздел может выступать в качестве руководство пользователя, так как является подробным объяснением работы практической реализации нашего метода. Задача автоматического установления авторского стиля была искусственно разделена нами на две части: программу сбора статистики и программу определения авторского стиля. Промежуточным звеном между ними является специализированная система построения нейронных сетей. Программа сбора статистики (TextStat) необходима для формирования тестовых данных и данных для исследования. Подготовленные с помощью нее данные служат основой для обучения нейронной сети, которое происходит в специализированной системе NeuroShell. Результатом работы системы является обученная нейронная сеть, которой пользуется программа установления авторского стиля (AUS). Именно она выполняет конечную задачу по соотнесению того или иного текста с авторским стилем. Она также включает в себя усеченную версию блока сбора статистики, который необходим ей для вынесения решения.

4.3.1. Сбор статистики и реализация лексического анализатора (TextStat)

Выделение этого блока в отдельную программу обусловлено необходимостью изучения данных для дальнейшего развития метода. Модуль сбора статистики TextStat состоит из двух основных частей: лексического анализатора и блока сбора статистики.

В круг задач лексического анализатора входят задачи выделения очередной лексемы из текста, отнесения лексемы к определенному классу (слово, цифра, знак препинания), маркировка лексем в соответствии с их типом (для лексем-слов – служебный тип: союз, частица, предлог, или другой тип; для лексем-знаков препинания – внутренние знаки:

запятая, тире, точка с запятой и др.; внешние знаки: точка, многоточие, восклицательный знак, восклицательный знак с многоточием, вопросительный знак и т.д.), а также определение границ предложений.

Блок сбора статистики занимается накоплением статистической информации в соответствии с поступлением данных от лексического анализатора. Для каждого набора данных он формирует характеристики и сохраняет их в выходном файле. Логика подсчета характеристик задается статически в алгоритме самой программы.

TextStat работает в терминах выборок, таким образом, файл на входе разбивается на равные фрагменты-выборки (длина которых задается пользователем), и характеристики подсчитываются для каждой выборки в отдельности. Взаимодействие с пользователем осуществляется через командную строку, в ней можно задать файл обработки, в том случае, если файл не задан, обрабатываются все файлы с расширением «txt». Программа работает в консоли и не имеет графической оболочки.

Результаты работы программы легко импортируются в файлы данных MatLab 6.0, а также в файлы .sta популярного пакета Statistica.

Программа реализована на языке C++ (с использованием возможностей объектно-ориентированного программирования) в среде Microsoft Visual C++ 6.0.

В нашем конкретном случае для тестирования возможностей программы были взяты произведения десяти классических писателей XIX-XX вв. из электронной библиотеки Мошкова [14]. Всего для обучения было обработано 42 текста общим объемом 21 Mb (см. Приложение 2), каждый текст разбивался на выборки по 6000 тысяч слов. В итоге было сформировано 482 обучающих примера, в последствии использующихся для обучения нейронной сети. Каждый пример состоял из набора характеристик, подсчитанной для данной выборки (входы нейронной сети), и фамилии писателя (выход нейронной сети). Таким образом, была сделана попытка обучить сеть различать авторский стиль 10 заранее заданных писателей.

4.3.2. Обучение нейронной сети на примерах (NeuroShell 2 Release 4)

Обучение нейронной сети производилось в NeuroShell 2 Release 4.0 – специализированной системе построения нейронных сетей. Была выбрана наименее простая и наиболее изученная топология сети – нейронная сеть прямого распространения. Сеть такого типа позволяет производить автоматическое разделение классов в многомерном пространстве на основе предложенной обучающей информации. Архитектура сети соответствовала рекомендациям данной системы: блок входных нейронов передавался на три блока нейронов скрытого слоя с разными пороговыми функциями (Гауссиан: $\exp(-x^2)$,

дополнение к Гауссиану: $1-\exp(-x^2)$, гиперболический тангенс) по 23 нейрона в каждом блоке (см. Рис. 0).

Для тренировки нейронной сети на выбранных нами данных, в среднем, понадобилось примерно 100 эпох и около 1 минуты времени на компьютере с процессором Celeron 600 Mhz. Сеть успешно решала задачу на тестовых данных с незначительными оговорками. На нескольких примерах можно было сделать неправильные выводы об авторстве, но при этом большинство (примерно девяносто процентов) тестов позволяло дать успешные оценки.

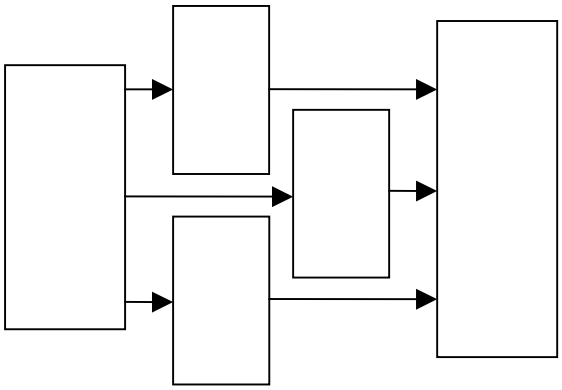

Рис. 0 Архитектура сети

4.3.3. Утилизация обученной нейронной сети (AUS)

Средствами системы NeuroShell был сгенерирован исходный код на языке C++, включающий в себя данные обученной нейронной сети. В результате интеграции полученного кода (являющегося блоком вынесения решений) с модулем накопления статистики была получена программа оценки авторского стиля – AUS (AUthor Style).

На входе через строку параметров программа получает путь к текстовому файлу, авторство которого необходимо установить, а на выходе формирует вектор из 10 значений, каждое из которых лежит в интервале от 0 до 1 и сопоставлено с одним из 10 выбранных нами авторов. Ноль – означает, что стиль данного текста не является стилем данного автора, единица – наоборот говорит о том, что стиль данного текста соответствует стилю того или иного автора. В идеале на выходе нейронной сети должен быть сформирован вектор со всеми нулевыми компонентами кроме одной, которая и указала бы автора. Дробные значения же теоретически являются ошибкой, указывающей либо неполную единицу, либо «разросшийся» ноль. Так как результатом работы сети на практике всегда является вектор с дробными значениями, то исследователю приходится разрабатывать свой алгоритм интерпретации. Например, можно сказать, что промежуточные значения говорят о «неуверенности» сети в вынесении решения. Величину значения можно интерпретировать как степень уверенности. В нашей программы отображался вектор, как есть, пользователь же мог выбрать метод интерпретации самостоятельно.

В ходе тестирования нейронной сети, обученной на текстах выбранных нами 10 писателей, было выявлено, что сеть выносит однозначные решения на известных ей текстах, которые использовались в процессе обучения, а также достаточно четко распознает

неизвестные тексты при условии, что они написаны одним из десяти авторов (см. Приложение 3, Результаты работы программы AUS). В случае, когда сети предлагается оценить авторский стиль текста, автор которого ей неизвестен, она также выносит решение и иногда можно сделать предпочтение в пользу того или иного писателя из 10 известных. Не исключено, что в таком случае можно говорить о близости авторских стилей, но абсолютно уверенным в этом быть нельзя, так как сеть обучалась распознавать авторский стиль заданных писателей, а не измерять авторский стиль в целом.

4.4. Недостатки и перспективы развития

Безусловно, такой подход не лишен недостатков. Прежде всего, он избыточен. Используются огромные массивы статистики основанных на слабо изученных характеристиках, большинство из которых могут не то что не пригодится, а даже помешать правильному распознаванию. Возможным путем решения данной проблемы может стать тщательная оценка значимости входов или прореживание сети известными методами.

К недостаткам данного метода можно также отнести отсутствие универсальности. Работа изначально ведется с фиксированным набором авторов, с априорно заданными языковыми особенностями, что сильно ограничивает сферу применения метода.

Направления по улучшению метода исходят из его недостатков. Возможным путем решения проблемы избыточности и несостоятельности входов может стать оценка значимости входов или прореживание сети известными методами. К настоящему моменту нами проводилась работа по отсеву имеющихся входов, отслеживанию корреляций между ними, выявление их значимости в задаче оценки авторского стиля, но с развитием метода, описанного в следующей главе, работа над данной проблемой была приостановлена.

Преодоление языковой зависимости, на наш взгляд, кроется в построении системы автоматического отбора характеристик, которая на основе достаточно большого числа текстов неизвестного ей языка пыталась бы строить различительные признаки. Как вариант, частично можно выбирать универсальные характеристики текста, языковые же особенности извлекать автоматически, посредством построения семантических карт [15]. Частично это проблема была разрешена в разработанном нами методе кластеризации текстов.

5. КЛАСТЕРИЗАЦИЯ ТЕКСТОВ

5.1. Постановка задачи

Реализация метода установления авторства текстов с фиксированным набором альтернатив даже в самом лучшем своем исполнении не решает всех проблем, с которыми может столкнуться пользователь. Предварительный отбор авторов сильно сковывает исследователя, не позволяя выйти на более высокий уровень анализа произведений. В этом смысле подход, основанный на кластеризации текстов, куда более универсален. Его идея заключается в следующем. Программе установления авторства последовательно предлагаются различные тексты. Следуя заложенному в ней алгоритму, она пытается произвести сортировку произведений в соответствии со своими выводами об их создателе. Сортировка производится в соответствии с принципами кластеризации – каждое произведение находит свое место в многомерном признаковом пространстве, где в свою очередь можно выделить группы (кластеры), обозначающие близость произведения к стилю того или иного автора. Отображение в признаковое пространство выполняется автоматически, то есть происходит классификация без учителя. Дальность расположения отдельных произведений, также как и удаленность кластеров друг от друга, соответствует отличию в авторском стиле произведений – чем дальше расположены объекты, тем сильнее отличие в стилях. Таким образом, можно не просто «узнавать» стиль конкретного автора, а соотносить авторский стиль текста со стилями других текстов.

5.2. Необходимость автоматического нахождения авторского инварианта

Смещение проблемы установления авторства в сторону кластеризации текстов, к сожалению, само по себе не решает проблем с выбором статистических характеристик, которые бы давали количественное описание текста. Алгоритм, производящий сортировку, произведений по-прежнему основывается на собранной статистике, зависящей в свою очередь от того набора характеристик, который исследователь посчитает наилучшим. Кроме того, проблема ставится еще более остро. Набор характеристик становится критическим значением, так как кластеризация производится автоматически, без учителя, без вмешательства человека. Если в случае с обучением нейронной сети на примерах, есть шанс подстройки даже к плохим данным[6], то при автоматической классификации плохие данные напрямую воздействуют на качество группировки. Отсюда следует, что необходим

[6] Первые версии нашей программы сбора статистики работали неверно, сеть же все равно обучалась и делала верные выводы, хоть и в гораздо меньшем числе случаем, чем сеть, обученная на правильных данных.

механизм, осуществляющий поиск характеристик, которые позволят сделать качественное разделение текстов по авторскому стилю. Так как в конечном итоге программе нужно оценивать, насколько хорош или плох очередной набор характеристик, то поиск должен производится с приготовленной заранее информацией. Что опять подводит к идее обучения с учителем – набор характеристик хорош в том случае, если алгоритм кластеризации на основе этого набора сделал правильное разделение кластеров, а кластеры разделены правильно в том случае, если в одном кластере находятся произведения одного автора. Эта оценка требует знаний о том, кто написал тот или иной текст. Казалось бы, вся идея автоматической кластеризации в таком случае сходит на «нет», но это не так. Поиск статистических характеристик может быть сделан единожды на большом числе примеров, а затем, если он «зарекомендует» себя на тестовых данных, число которых исследователь посчитает достаточным, им можно будет пользоваться для классификации любого количество текстов.

В отличие от метода с фиксированным набором альтернатив, где выбор характеристик так и остается на совести исследователя, здесь число свободных параметров алгоритма уменьшается. Среди них остается лишь число и характер примеров, на множестве которых производится поиск универсальных характеристик, сами же характеристики являются производными и находятся автоматически. Конечно, это все еще серьезная проблема, требующая доказательств (сколько текстов достаточно, чтобы найденные характеристики можно было считать универсальными? сколько и какие авторы должны быть среди выборочных примеров? и т.д.), но, во всяком случае, ее можно попытаться решить количественно. Чем больше текстов и всевозможных авторов, тем больше шансов набору характеристик стать универсальным, в то время как при ручном выборе характеристик данным подходом воспользоваться нельзя – число всех возможных характеристик куда больше количества всех имеющихся текстов.

Прежде чем приступить к практическому описанию предлагаемого нами метода необходимо разрешить еще несколько принципиальных вопросов. Если с оценкой качества группировки произведений интуитивно все должно быть понятно (подробное описание можно найти далее), то сам поиск характеристик требует пояснений. Совершенно ясно, что поиск должен иметь глобальный характер. Более того, пространство различных наборов характеристик имеет неопределенно высокую сложность, поэтому может иметь бесконечное количество локальных экстремумов. Также необходимо подчеркнуть, что каждая точка пространства не просто отдельная характеристика текста (например, число появлений в нем служебного слова «как»), а целый набор характеристик (например, число появлений слова «как», число слов длины 5, число союзов «что» и средняя длина предложения). При этом другой набор, отличный от первого на одну единственную характеристику, может

располагаться сколь угодно далеко от своего «собрата». У нас нет метрики пространства набора характеристик. Перемена одной единственной характеристики – это шаг неопределенной длины в неопределенном направлении. Все что известно об этом пространстве, так это расположение набора характеристик в нем, а скорее даже уровень на котором располагается данный набор в пространстве, то есть полезность в терминах алгоритма кластеризации текстов.

Чтобы организовать поиск, нужно определить метрику, задать минимальный шаг, пусть даже это будет шаг в бесконечность. Методов, которые бы работали с таким минимумом информации и подобными ограничениями, не много. Наиболее очевидный – это полный перебор, начинающий с минимального набора характеристик и последовательно изменяющий его в сторону бесконечности. Подобный подход с учетом колоссального объема анализируемой информации не годится: он может никогда не достигнуть цели. По той же причине не подходят методы последовательного приближения к цели (даже, если шаг приближения будет варьироваться, нас ожидает множество проблем в правильной организации поиска). Простой перебор рандомизированных решений не обладает сходимостью и может также не найти ничего ценного. В нашем случае вновь была произведена попытка обращения к методам искусственного интеллекта, а в частности к генетическим алгоритмам.

5.3. Генетические алгоритмы как метод глобального поиска: идея, терминология и обоснование эффективности

Генетические алгоритмы – это адаптивные методы поиска, последнее время часто используемые для решения задач функциональной оптимизации. Они основаны на идее генетических преобразований биологических организмов. Популяции, подчиняющиеся законам естественного отбора («выживает наиболее приспособленный»), развиваются в течение нескольких поколений. Подражая этому процессу генетические алгоритмы способны «развивать» решения реальных задач, если те соответствующим образом закодированы.

В терминах генетических алгоритмов закодированная строка называется «генотипом» (genotype) или «хромосомой» (chromosome), а породивший ее объект «фенотипом» (phenotype). Каждому генотипу соответствует оценка качества или «выживаемости», определяемая фитнес-функцией. На основании этой оценки происходит отбор получаемых особей из популяции (набора генотипов).

В рамках каждого поколения происходит получение «потомства» из существующих особей путем применения генетических операторов (кроссовингер, мутация, инверсия) и дальнейший отбор наиболее выживаемых особей (селекция). Процесс прекращается при

достижении необходимой точности решения или остановке прогресса с каждой последующей популяцией.

Роль генетических операторов в процессе поиска – модификация существующих решений для получения новых, возможно более качественных. Изменение решений предполагает целенаправленность, которая заключается в ускорении процесса сходимости и поиске лучшего решения. Выбор генетических операторов и их параметров должен максимально соответствовать поставленной задаче, иначе допустимо ухудшение результатов поиска.

Идеи двух основных генетических операторов заключаются в следующем. Мутация – это случайное изменение маленького участка информации в генетическом коде. Кроссовингер имитирует скрещивание биологических организмов. Многие исследователи считают оператора кроссовингера ключевым в генетических алгоритмах, так как именно он придает характер целенаправленности случайному поиску. При правильно выбранном коде и удачно подобранной модификации кроссовингера новообразованные решения будут включать в себя лучшие качества своих предков, а каждое новое поколение будет наращивать свою выживаемость. После преобразования существующих решений генетическими операторами из старых и новых решений выбираются самые лучшие, а также те, которые в дальнейшем могут дать лучшие решения.

Преимущества генетических алгоритмов перед другими методами глобального поиска заключаются в их особенной работе с пространствами, которые обладают множеством локальных экстремумов. Происходит поиск с кажущимися хаотическими движениями по области определения решений, но, как уже говорилось выше, процесс поиска целенаправлен. Эффективность работы генетических алгоритмов можно доказать, обращаясь к схемной теории обработки решений. Труды Голдберга [16] и Голланда [17] показывают, что в генетическом поиске на самом деле происходит работа не непосредственно с решениями, а так называемыми шимами, представляющими собой шаблоны подобия между генетическими строками. В ходе работы алгоритма осуществляется перебор не всех возможных решений кодового пространства, а гиперплоскостей в областях поиска с высокой приспособленностью. Гиперплоскости представляют собой множество схожих строк.

Шима (schema) – это строка определенной длины (равной длине строк популяции), состоящая из каких-либо возможных знаков кодового пространства и неопределенных символов [18]. Неопределенные символы обозначают возможность нахождения любого символа кодового алфавита в данной позиции. Для бинарного пространства кодов, например, шима "0*10*" будет шаблоном для следующих строк: "0110", "01101", "00100", 00101". Строки, для которых шима является шаблоном, называются примерами шимы.

Выделяют два основных свойства шим: порядок и определенная длина. Порядком шимы называют число определенных символов в шиме (в нашем примере порядок равен 3). Определенной длиной называется расстояние между крайними определенными символами в шиме (для "0*10*" определенная длина равна 4). Каждая строка в популяции является примером двух в степени размера популяции шим.

Понятие строящих блоков впервые ввел Голдберг в 1989 году. Строящие блоки – это шимы, обладающие высокой приспособленностью, низким порядком и короткой определенной длиной. Шима имеет приспособленность равную средней выживаемости ее примеров.

Процесс отбора особей чаще оставляет строки с более высокой приспособленностью. Кроссовингер почти не разрушает строки с короткой определенной длинной, а мутация слабо разрушает шимы с низким порядком. Таким образом, строящие блоки имеют наиболее высокие шансы переходить из поколения в поколение.

Голланд показал, что, явным образом обрабатывая N строк на каждом поколении, генетический алгоритм в тоже время неявно перебирает порядка N^3 таких коротких шим низкого порядка и с высокой приспособленностью (полезных шим) [18]. Такое явление стали называть неявным параллелизмом.

С помощью теоремы шим Голдберг показал, что строящие блоки растут по экспоненте, а шимы с приспособленностью ниже средней распадаются с той же скоростью. Гипотезу Голдберга о строящих блоках можно сформулировать, как «строящие блоки объединяются, чтобы сформировать лучшие строки», при этом рекомбинация и экспоненциальный рост являются гарантом получения лучших решений. Однако, такая интерпретация теоремы не совсем верна, так как некоторые схемы не отбирают воспроизводство, и число различных схем для рассмотрения зависит от размера совокупности [19]. Исследования показали, что генетические алгоритмы дают гарантированно лучшее решение лишь для простых оптимизационных задач. Для решения сложных задач необходима сходимость с более строгим управлением, но методология такой сходимости до сих пор не выработана.

Тем не менее, генетический поиск со своим скачкообразным поиском решений наилучшим образом приспособлен к задаче поиска наборов характеристик, предназначенных для кластеризации текстов. Помимо скрытого параллелизма для него можно разработать механизм явного распараллеливания перебора признаков (для компьютеров с многопроцессорной архитектурой, либо кластерных систем), что очень актуально с учетом большой трудоемкости фитнес-функции оценки набора характеристик (подробнее см. п.5.4.2 Реализация целевой функции, ее варианты). Кодирование же набора признаков генетической строкой было для нас наилучшей подсказкой организации перебора.

5.4. Этапы разрешения проблемы

С определением метода поиска набора характеристик, устройство программы кластеризации текстов выстроилось в единую схему, реализацию которой осуществлялось нами поэтапно. В процессе прохождения очередного этапа формировались слои теории, в которые вошли все наши мысли относительно вариантов исполнения каждого модуля, тонкости и принципы реализации блоков, а также термины, которые были необходимы для описания всех реализованных функций. Данный раздел содержит теорию, которой достаточно для понимания принципов работы алгоритма. Для более детального описания смотрите: Приложение 5 Руководство программиста.

5.4.1. Построение кода характеристик. Терминология

Выбор кодировки в генетических алгоритмах играет решающую роль. В случае удачно выбранной кодировки более качественные решения находятся за гораздо меньший промежуток времени. Важно, чтобы все возможные варианты кодовой строки полностью покрывали пространство решений, а лучше даже совпадали с ним. Если кодирование предполагает порождение заранее неверных вариантов, которые не удовлетворяют ограничениям задачи, то есть вероятность, что генетические алгоритмы будут осуществлять поиск не лучших решений, а вообще возможных (например, среди десяти полученных решений лишь два удовлетворяют условиям задачи). А в случае неполного покрытия пространства решений кодовым пространством некоторые решения вообще не будут рассмотрены.

Таким образом, выбор кода – непростая задача. Он должен быть достаточно емким, чтобы отражать все возможное признаковое пространство, компактным, чтобы не нужно было каждый раз анализировать громоздкий код, универсальным, чтобы работать с любым количеством символом (и любым языком в потенциале), а также полным (чтобы изменение или трансформация любой части кода образовывала также правильный код). В нашей первой программе работающей с фиксированным количеством авторов сбор характеристик задавался явным образом (классическими операторами циклов, присвоений и арифметических операций) и без перекомпиляции программы изменить его было невозможно. Для работы же генетического алгоритма характеристики должны задаваться в генетической строке, а, следовательно, одновременно с ее структурой нужно было продумывать механизм сбора статистики (то есть фактически расшифровки генотипа). Первоначально проводилась работа над задачей построения кода, позволяющего организовать сбор той же статистики, которая раньше была статически заложена в программе. В результате получился более сильный код, хотя некоторые характеристики из

программы установления авторства с фиксированным набором альтернатив в нем реализовать невозможно.

Придуманный нами код всегда имеет четную длину. Числа на нечетных позициях являются номерами атомов, тогда как следующее за каждым из них число на четной позиции их атрибутом. Прежде чем программа сможет обрабатывать такой код, ей необходимо загрузить атомы, а следовательно каждый код привязан к своему набору атомов. Набор атомов – это область определения геномов на нечетных позициях (геномов-атомов).

Существует базовый набор атомов, который реализуется в виде списка и включает в себя все возможные атомы, заложенные в программе. В нашем случае это вся ASCII таблица символов, некоторые составные знаки препинания (многоточие, вопрос, восклицательный знак с многоточием и др.), и служебные атомы. При необходимости программа легко расширяется для работы с символами Unicode.

Рабочий список атомов – это массив, содержащий те атомы, которые участвуют в формировании конкретной генетической строки, либо атомы, привязанные к уже сформированной генетической строке. Рабочий список атомов состоит из двух частей: подмножества базового набора атомов и группы макроатомов. Макроатомы – это особый класс атомов, которые являются склейкой базовых атомов. Например, вместо того, чтобы кодировать в строке каждую букву слова «как» (на что уйдет три гена плюс три атрибута к ним), можно закодировать это одним геном и одним атрибутом. Только предварительно нужно загрузить макроатом слова «как».

Служебных атомов немного. Один из самых важных – разделитель характеристик. Если он встречается в строке, то это сигнализирует начало новой характеристики. Вся генетическая строка была названа нами вариантом, потому что она является всего лишь вариантом набора характеристик. Таким образом, вариант может включать несколько характеристик, разделенных между собой кодом разделителя характеристик.

Одна характеристика может содержать в себе несколько шаблонов для слов, разделенных между собой служебным атомом – разделителем слов. Эти слова считаются рядом стоящими, и в одной характеристике может содержаться сколько угодно рядом стоящих слов, образующих цепочку.

Генетическая строка (вариант) задает собой шаблон сбора статистики. Каждая характеристика в ней в результате обработки одного текста дает вектор-столбец значений (по одному на каждую выборку текста, которая участвует в анализе). Отсюда следует, что весь вариант задает шаблон сбора статистики, результатом которого является массив векторов-столбцов с числом столбцов равным числу характеристик в варианте, и с числом строк равным количеству выборок, которые участвуют в анализе. Шаблон сбора статистики – всего

лишь образное выражение, на деле же каждая характеристика – это булево выражение, которое последовательно применяется к каждому слову текста. Если условие выполняется (будем рассматривать пока только одну выборку), то счетчик характеристики увеличивается на единицу. Когда характеристика содержит несколько слов, то к текущему слову текста применяется первое слово характеристики, к следующему после текущего – второе, и так далее. Счетчик характеристики увеличивается только в том случае, если все булевы выражения для всех слов характеристики были истинными. После сбора статистики получается массив векторов-столбцов с накопленными значениями характеристик, каждый элемент которого делим на длину выборки.

Помимо разделителей есть еще один служебный атом – это длина слова. Его атрибут – это сама величина длины, атрибуты же других атомов (букв) указывают на их месторасположение в слове. Например, строка «1 2» (при условии, что атом с номером 1 это буква «А») интерпретируется как «буква А, стоящая на втором месте в слове». Если две буквы характеристики стоят на одном и том же месте, работает правило «ИЛИ» (либо одна буква, либо другая – не важно какая именно). У атомов разделителей атрибуты равны нулю.

Попробуем дать интерпретацию произвольного варианта. Условимся, что атомы с 1 по 33 – это буквы русского алфавита («А, Б… Я»), «-1» – это разделитель характеристик, «-2» – разделитель слов, «34» – атом длины слова. Тогда строка «1 2 5 4 -2 0 4 2 5 2 -1 0 34 3 2 1» расшифровывается следующим образом. Первая характеристика: подсчитать все слова, у которых вторая буква «А», а четвертая буква «Д», стоящие справа от слова, у которого вторая буква «Г» либо «Д» (здесь цифры «1 2 4 5» – это шаблон первого слова, «4 2 5 2» – второго). Вторая характеристика: подсчитать все слова длины 3, у которых первая буква «Б».

Как видно из примера, основное преимущество такого кода в том, что любой вариант может получить свою языковую интерпретацию. Конечно, она получается слишком громоздкая, но при необходимости разобраться в ней можно. Помимо этого, нетрудно сделать графическое представление варианта. Например, вышеуказанный вариант мог выглядеть следующим образом: _А__Д… _Г(Д)…, Б__.

В генетическом поиске участвуют только атомы из рабочего списка, поэтому можно задавать произвольный вид строки. Формат кода получился универсальным, так как позволяет задавать любой набор символов для перебора, а значит и любой язык, и емким, раз с его помощью можно задавать сколь угодно сложные описания, включающиеся в себя шаблоны не только целых предложений, но и кусков текста.

Корректность строки определяется тем, нет ли на четных позициях номеров несуществующих атомов, и тем, все ли аргументы атомов имеют положительное значение. Длина генетической строки переменная и может варьироваться в границах от двух до

бесконечности (параметры нижней и верхней границы можно задавать по желанию), но она должна всегда оставаться четной.

5.4.2. Реализация целевой функции

Существенной частью генетического алгоритма является выбор целевой функции (фитнес-функции). Фитнес-функция отвечает за оценку решений. Ее задача определить качество решения исходя из вида конкретной хромосомы. Первая часть подсчета фитнес-функции любого варианта в нашем конкретном случае поиска набора характеристик заключается в формировании массива векторов-столбцов статистики на основе того шаблона, что дает этот вариант. Разбор генетической строки и описание примерного алгоритма сбора статистики было дано в предыдущем разделе. Осталось раскрыть, каким образом массив векторов-столбцов накопленной статистики обращается в единственное число – значение фитнес-функции хромосомы.

Нами было найдено простое и эффективное решение этой задачи. Это одно из немногих возможных решений, и не исключено, что оно не самое оптимальное. Фитнес-функция в данном случае вычисляется как отношение дисперсии классов авторов к средней дисперсии внутри классов авторов:

$$\frac{1}{n-1}\sum_{i=1}^{n} M(\overline{A},\overline{A}_i)^2 \Big/ \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{k-1}\sum_{t=1}^{k_i} M(\overline{A}_i, A_{it})^2\right)$$, где

n – число различных авторов;

k_i – число выборок в текстах у i-го автора;

A_{it} – вектор координат t-го текста i-го автора в стилеметрическом пространстве для найденного набора признаков (имеет размерность равную количеству характеристик);

$\overline{A}_i$ – выборочное среднее координат выборок по текстам i-го автора в стилеметрическом пространстве для найденного набора признаков (имеет размерность равную количеству характеристик);

$\overline{A}$ – выборочное среднее выборочных средних координат выборок по текстам всех писателей в стилеметрическом пространстве для найденного набора признаков (имеет размерность равную количеству характеристик);

$M()$ – функция нахождения расстояния между векторами.

Результатом сбора статистики является массив векторов-столбцов, по столбцу на каждую характеристику, и по строке на каждый пример (выборку). То есть, фактически строится многомерное признаковое пространство (не путать с пространством набора характеристик!). Перед запуском же алгоритма выбираются тексты, на которых производится обучение, причем автор каждого заранее известен. Таким образом, какие-то строки пространства можно отнести к одному автору (они соответствуют выборкам из произведений этого автора), какие-то к другому. Каждая строка, соответствующая конкретной выборке, является точкой в многомерном признаковом пространстве. Исходный вариант (генотип) является порождающим для этого пространства. Значение фитнес-функции варианта, породившего данное пространство, тем выше, чем лучше группируются точки-выборки одного писателя, и чем дальше группы (кластеры – отсюда понятие «кластеризация текстов») разных писателей расположены друг от друга. Поэтому значение фитнес-функции вычисляется как отношение дисперсии кластеров писателей (разброс кластеров в признаковом пространстве – чем больше, тем лучше) к средней дисперсии внутри классов авторов (разброс точек-выборок одного писателя – чем меньше, тем лучше).

Помимо обозначенного нами способа, были рассмотрены также его модификации, учитывающие последующее отображение текстов с учетом кластеризации (см. раздел 5.4.4. Картографическое представление результатов: PCA и SOM), в случае которого первоначально производится редукция многомерного признакового пространства до двумерного, и только после этого оценивается расположение кластеров.

5.4.3. Реализация генетического поиска на базе библиотеки GALib

Программирование ядра генетического алгоритма задача не простая и по сути ненужная. Вместо того чтобы разрабатывать с нуля всю библиотеку классов, нами был осуществлен поиск подходящей библиотеки в Интернет, и была найдена та, которая полностью удовлетворяет нашим требованиям.

Библиотека GALib – это полнофункциональная библиотека объектов и методов для разработки генетических алгоритмов. Она включает в себя типы данных, различные классы генотипов, генетических алгоритмов, популяций, схем селекций, классов сбора и накопления статистики по работе алгоритма и классы генерации случайных чисел. Все права на распространение принадлежат Массачусетскому Технологическому Институту и автору программы – Мэтью Волу (Matthew Wall). Библиотеку можно использовать и распространять сторонним в лицам в целях разработки некоммерческих программ.

В идеале, программисту, пользующемуся библиотекой GALib, достаточно выбрать класс генетических алгоритмов и генотипов, которые нужны для решения его задачи,

установить нужные параметры, и написать фитнес-функцию. В нашем случае все получилось несколько сложнее. Для правильной работы генетического алгоритма пришлось переписать процедуру инициализации генотипов, а также два главных генетических оператора – кроссовингер и мутацию. В библиотеке заложены возможности переопределения базовых функций без необходимости написания наследственного класса.

В качестве типа хромосомы (генотипа) был выбран класс GA1DArrayGenome с целочисленными элементами (integer). Это масштабируемый массив, все функции для работы с которым заложены в библиотеке GALib. Что касается класса самого генетического алгоритма, то нами был выбран класс GASteadyStateGA. Это алгоритм, использующий перекрывающиеся популяции с задаваемым пользователем коэффициентом перекрытия. Это означает, что очередная популяция содержит заданный процент особей из предыдущей популяции, а остальные члены набираются из числа образованных в результате работы генетических операторов.

Часть опций генетического алгоритма в нашей реализации задается прямо в программе, часть может быть изменена в диалоге, управляющем запуском генетического алгоритма.

Помимо выбора классов, о котором было написано выше, необходимо подробно расписать переопределенные нами операторы генерации новых генотипов и фитнес-функцию.

Первое, с чем пришлось столкнуться в ходе работы с GALib, помимо выбора классов и настройки параметров, была реализация фитнес-функции. Не считая подстройки к данным генетической библиотеке и отладки, с ней не возникло проблем – все этапы оценки качества генетической строки были продуманы и реализованы нами заранее. Вначале происходила оценка корректности варианта (в случае ошибок в строке возвращалось нулевое значение фитнес-функции), далее следовало формирование массива векторов-столбцов на основе сбора статистики, и на последнем этапе оценивалось расположение кластеров писателей в многомерном пространстве признаков.

Следующее, без чего было не обойтись – инициализация генотипов. Процедура инициализации по умолчанию вырабатывала произвольные значения, большинство из которых были нулевые. Попытка реализовать множество значений геномов в генотипе переходом от класса GA1DArrayGenome к классу GA1DArrayAlleleGenome, где можно задавать базовый набор геномов, не увенчалась успехом. В нашем случае геномы на четных позициях (атомы) могут иметь один набор значений, а геномы на нечетных позициях (аргументы) совершенно другой. Пришлось задавать ограничения в каждой из следующих процедур: инициализация, мутация и кроссовингер. В ходе инициализации генотипов

выбирается произвольная длина строки и каждому геному в ней присваивается случайное значение (удовлетворяющее ограничениям области определения). Так геномы на нечетных позициях (атомы) могут иметь значение только номеров атомов из рабочего списка, а геномы на четных позициях (аргументы) – возможные позиции букв в слове (от 1 до 10, слова с длиной больше 10 не учитываются).

Процедура мутации заключается в случайном изменении отдельного генома строки. В цикле по всем геномам строки алгоритм мутации с низкой вероятностью меняет отдельный геном на другой, удовлетворяющий ограничениям области определения. Как и в случае инициализации, алгоритм, по умолчанию заложенный в библиотеке, не учитывал ограничений области определения и тем более не мог различить четные и нечетные позиции.

С кроссовингером дело обстояло несколько иначе. Скрещивание не работает с отдельными геномами, а вырабатывает новые генотипы на основе двух старых. Процедура по умолчанию не учитывала, что строка всегда должна быть четной длины, а точки разбиения не должны разделять атом с его характеристикой (то есть разбиение всегда происходит по четным позициям).

Помимо операторов генерации новых генотипов в генетическом поиске есть оператор селекции, который осуществляет выбор генотипов из популяции для их скрещивания, либо, если алгоритм допускает перекрытие, для выбора тех генотипов, которые перейдут в следующую популяцию. Существует множество алгоритмов селекции (отбора), самые популярные из которых реализованы также и в GALib: рейтинговый отбор (GARankSelector), вероятностное колесо (GARouletteWheelSelector) и турнирный отбор (GATournamentSelector).

Рейтинговый (элитный) отбор предполагает сортировку полученных и существующих ранее особей и выбор «верхушки» с отсевом решений, не попавших в число лучших. Такой подход обеспечивает быструю сходимость, но не дает необходимую гибкость. Тем не менее, его использование целесообразно во многих задачах.

Вероятностное колесо ориентировано на гибкость поиска. Совокупность всех полученных решений представляется в виде рулетки, поля которой – отдельные решения. Размер каждого поля пропорционален значению фитнес-функции решения. Отбор заключается в случайном вращении колеса и последовательном выборе наиболее удачливых решений. Очевидно, что особи с высокой выживаемостью имеют большие шансы быть выбранными и войти в следующее поколение. В нашем случае был выбран именно этот метод, но в перспективе нами планируется предоставлять выбор пользователю.

Турнирный отбор является методом, совместивший в себе идеи элитного отбора и вероятностного колеса. Для выбора лучших k строк организуют k турниров, в процессе

которых выбирают произвольное число особей из популяции и среди них отбирают одну с наивысшим фитнес-показателем.

Когда все необходимые функции (кроссовингер, мутация и фитнес) были реализованы, встала задача оптимизации полученного алгоритма. Дело в том, что для поиска универсального набора характеристик требуется работа с большим количеством текстов. Как и в случае с установлением авторства при фиксированном наборе альтернатив, были взяты же 10 писатели и их 40 текстов (см. Приложение 2), что составляет 21 Мб информации. В ходе генетического поиска приходится анализировать этот объем информации не один раз – фактически каждый новый вариант требует нового просмотра всех загруженных текстов. Допустим, размер популяции N генотипов (как правило, N=20). Таким образом, для создания новой популяции требуется N-1 просмотров (самый лучший индивидуум мигрирует из предыдущей популяции). Практика показала, что заметное улучшение лучшего индивидуума происходит каждые 10 поколений. Для нахождения достойного решения нужна не одна сотня поколений.

Что касается трудоемкости просмотра, то вычислить ее однозначно не получится. Операторы кроссовингер и мутация основаны на случайных процессах – каждый из них вступает в действие с определенной вероятностью, причем длина очередного генотипа также случайна. Трудоемкость подсчета одного генотипа зависит от его длины и содержания. Если длина варианта равна L, число слов в выборке – M, а число выборок во всех текстах – K. То трудоемкость сбора статистики приблизительно равна $L \cdot M \cdot K$. Трудоемкость подсчета значения фитнес-функции на основе матрицы векторов-столбцов характеристик (при числе характеристик равном H и числе авторов – A) равна $2 \cdot A \cdot H + A^2$. Таким образом, общая трудоемкость подсчета фитнес-функции одного варианта равна:

$$L \cdot M \cdot K + (2 \cdot A \cdot H + A^2)$$

Трудоемкость подсчета фитнес-функции для всех вариантов популяции, равна трудоемкости подсчета фитнес-функции каждого варианта. Количество поколений для нахождения хорошего набора характеристик, который позволил бы разделить тексты выбранных писателей (а в потенциале тексты и других авторов) неизвестно и зависит от материала для обучения, а также в определенной доле «везения» в ходе случайных процессов в генетическом алгоритме.

Для ускорения работы генетического поиска нами была реализован механизм кэширования отдельных характеристик. Так как в процессе применения кроссовингера многие уже подсчитанные характеристики могут появляться во вновь сформированных генотипах, то считать их несколько раз не имеет смысла. Нами был написан механизм,

сохраняющий (кэширующий) значения уже подсчитанных характеристик (они представляют собой вектор столбец – по одному значению на каждую выборку) и в случае их повторного появления использующий уже подсчитанные величины. Так как число различных характеристик может быть очень большим, то хранить все найденные характеристики невозможно. В нашей программе каждая характеристика в КЭШе имеет «срок жизни» по истечению которого, она уничтожается. Срок жизни продлевается, если на характеристику есть спрос. Такая система позволяет в значительной мере ускорить генетический поиск и находить приемлемые решения за меньший промежуток времени.

5.4.4. Картографическое представление результатов: PCA и SOM

Допустим, нужный набор характеристик найден. Он обладает потенциалом кластеризации текстов по авторству. Как отобразить или показать это? Конечному пользователю нужен инструмент, позволяющий организовать наглядную работу с текстами. Без специальных процедур результатом обработки текста на основе найденного набора характеристик является многомерный массив данных, либо значение фитнес-функции, которое может помочь генетическому поиску, но не человеку, решившему исследовать авторство заданного набора текстов.

Для того чтобы отобразить многомерный массив данных на плоскость, для демонстрации на экране компьютера, существуют специальные методы визуализации и редукции многомерных пространств. Среди них метод главных компонент (PCA) и самоорганизующиеся карты Кохонена (SOM).

Метод главных компонент [20] является частным и более простым случаем метода главных факторов, хотя в достаточной мере эффективным. На практике он, как правило, служит способом задания начального приближения для множества сложных методов (в том числе для SOM- и GTM-сетей). В нем изначально предполагается возможность полного разложения дисперсии элементарных признаков (а значит взаимодействия между этими признаками). Т.е. считается, что изучаемое явление полностью объясняемо скрытыми факторами.

Поиск главных компонент сводится к последовательному выделению первой главной компоненты, которая объясняет наибольшую дисперсию данных, второй главной компоненты, учитывающую наибольшую часть из оставшейся дисперсии, и т.д. На практике все компоненты получаются одновременно. Первоначально их число равно числу элементарных признаков. Определение значимости же каждой компоненты происходит на этапе оценки новой системы координат.

В нашем случае необходимо выделить лишь две главные компоненты независимо от того, какой процент дисперсии данных они объясняют. С помощью метода главных компонент можно преобразовать первоначальное многомерное признаковое пространство в двумерное. Таким образом, каждой выборке будет соответствовать вектор координат в двумерном пространстве (X, Y). После преобразования можно отобразить все множество выборок на экране монитора. Подобное представление позволит визуализировать полученные в ходе поиска набора характеристик кластеры. В идеале тексты каждого писателя будут образовывать группы, достаточно удаленные друг от друга, чтобы не происходила путаница между ними.

Самоорганизующиеся карты Кохонена – это вид нейронных сетей, обучающихся без учителя и предназначенных для обработки больших массивов многомерной информации. За последние годы это направление одно из наиболее развивающихся. С помощью SOM-сетей решаются многие проблемы классификации, обработки естественного языка [21], изображений [22], тестирования и обучения. Несмотря на широкое использование, SOM-сетям не хватает теоретической обоснованности – они опираются в основном эмпирические результаты.

SOM-сети являются развитием нейронных сетей Кохонена, обучающихся по принципу «победитель забирает все». В сетях Кохонена, по топологии относящимся к сетям прямого распространения, после воздействия примером выборки на вход только один нейрон возбуждается, остальные подавляются. Сеть Кохонена осуществляет классификацию входных векторов в группы схожих, подстраивая веса так, чтобы входные образы, принадлежащие одному классу, активизировали один и тот же выходной нейрон [23]. Для работы алгоритма важно, чтобы входные векторы были предварительно нормализованы. Нормализация происходит точно так же, как и в методах факторного анализа:

$$z_{ij} = \frac{x_{ij} - \overline{x_i}}{\sigma_j}.$$

Обучение сетей Кохонена происходит следующим образом. На вход подается нормализованный вектор и вычисляется расстояние между ним и вектором весов каждого нейрона. Затем производят аккредитацию, подстройку весов выбранного нейрона по определенным правилам. Подстройка весов сводится к минимизации разницы между входным вектором и вектором весов выбранного нейрона.

Идея SOM-сетей появилась в результате обследования самоорганизации, происходящей в сенсорных отделах человеческого мозга. Входные данные отражаются на двумерной карте нейронов, соединенных между собой двусторонними связями. Классическая SOM-модель

состоит из множества узлов (нейронов), упорядоченных на регулярной решетке (2-х или 3-х мерного пространства). Несмотря на внешне существенные отличия, SOM-сети утилизируют идею нейронных сетей Кохонена. Здесь обучение также происходит по принципу «нейрона-победителя». Только в SOM-сетях «победитель» забирает не все, а лишь часть изменений. Остальное распределяется по его соседям на карте. Регулярная решетка (карта) в этих сетях необходима для формирования пространственного содержания. С помощью нее выбираются соседи победившего нейрона.

Таким образом, SOM-сеть пытается отобразить структуру многомерного пространства признаков на двумерной карте (латентном пространстве) с наименьшей потерей информации (см. **Ошибка! Источник ссылки не найден.**2), и также может быть использована для визуализации классов авторов, выделенных в ходе поиска оптимального набора характеристик.

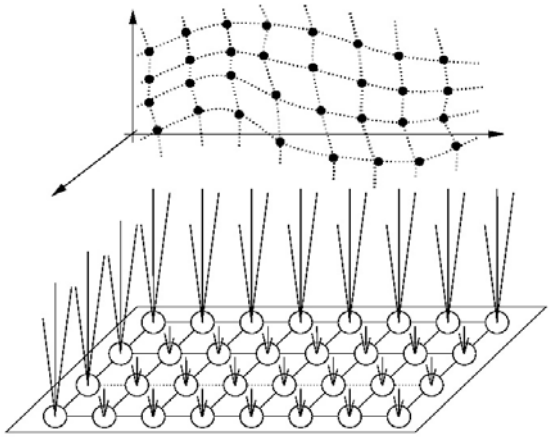

Рис. 2. Идея SOM-сетей

Распределение классов писателей в полученном двумерном пространстве признаков обладает множеством интересных свойств. Например, степень удаленности кластеров отражает степень различия в авторских стилях. Разброс произведений одного писателя внутри его кластера говорит о разных этапах творчества. Все это лишь малая толика той информации, которая может быть получена посредством картографического представления кластеров текстов.

5.4.5. Решающая функция

Пользователь не всегда довольствуется визуализацией данных. Иногда требуется дать четкий ответ на вопрос (например, при добавлении очередного текста на карту необходимо сказать к какому из отображенных на карте писателей больше тяготеет его стиль), либо автоматизировать полученные результаты (классический пример – поисковая система в Интернет). В таком случае необходима решающая функция, которая бы проанализировав карту (либо многомерное пространство), даст исчерпывающий вопрос на ответ. Число и вид подобных функций бесконечно (все зависит от требований к конечному результату), поэтому реализация решающей функции не входит в круг задач настоящей работы.

5.5. Достоинства и недостатки подхода

Предложенный метод уникален (поиски похожих методов в Интернет не увенчались успехом) и обладает рядом преимуществ по сравнению с методом установления авторства с

заданным набором альтернатив. Прежде всего, благодаря атомарной структуре генетической строки, он универсален. Описанный поиск набора характеристик позволяет работать с любым набором символов, а значит с любым знаковым языком. Теоретически метод может обрабатывать любую информацию (не только тексты) – например, достаточно переписать блок лексического анализатора и можно устанавливать авторство компьютерных программ.

В случае удачного нахождения универсального набора характеристик можно обрабатывать любое число авторов и текстов. Достаточно постоянно модифицировать карту, добавляя новые произведения и оценивать, как они взаимодействуют с ранее присутствующими.

С помощью универсального набора характеристик можно достаточно быстро обрабатывать большие массивы информации. Данная возможность может быть использована в поиске по Интернет, либо организации электронных библиотек. Метод может производить автоматическую кластеризацию текстов неизвестных ему авторов.

Найденным наборам характеристик, как уже было раскрыто выше, можно давать интерпретацию на простом человеческом языке, либо в виде шаблонов. Таким образом, программа способна вырабатывать правила, соперничающие с описаниями стилистов. Только в отличие от правил стилистов они будут действительно уникальными для автора (статистически обоснованы).

Одним из серьезных недостатков метода является невозможность прогнозирования успешного результата. Генетический поиск на заданном наборе текстов может никогда не найти хороший вариант для разделения характеристик. Нет никакого критерия того, в правильном ли направлении движется поиск, верно ли он делает скачки, нужную ли скапливает информацию об исследуемом пространстве. Исследователь сам должен производить мониторинг поиска и следить за всеми «поворотами событий». Кроме того, нет механизмов, определяющих, сколько времени осталось до конца работы алгоритма, до того момента, когда дальнейший поиск не принесет своих результатов. Надеемся, что в процессе эксплуатации данного метода и дальнейшем его исследовании, такие критерии будут найдены, и проблема не будет стоять столь остро.

Другой проблемой метода является его трудоемкость. Число загруженных текстов, которое напрямую влияет на качество поиска, требует больших ресурсов от вычислительной системы (большой объем памяти и мощный процессор). Для нахождения по-настоящему универсальных характеристик необходимо обработать не один десяток мегабайт текстов, чтобы можно было с уверенностью заявить об их универсальности. Данная проблема может быть решена в процессе анализа шагов алгоритма. Не исключено, что некоторые фрагменты программы можно будет оптимизировать, где-то использовать уже накопленную статистику

(по аналогии с кэшированием характеристик), где-то продвинуться сразу на несколько шагов вперед. Что касается памяти, то на данный момент нами специально не рассматривался вопрос ее оптимального распределения, но потенциально возможно разработать вариант с меньшей загрузкой.

5.6. Функциональные возможности программы TextScout (руководство пользователя)

В данном разделе содержаться сведения о нашей реализации метода кластеризации текстов на основе автоматически установленного набора характеристик. Он также является руководством пользователя, так как включает обзор всех функций, содержащихся в главном меню, описание интерфейса и возможностей программы. Наша реализация вышеописанного метода включает в себя все основные этапы поиска.

5.6.1. Общий вид программы

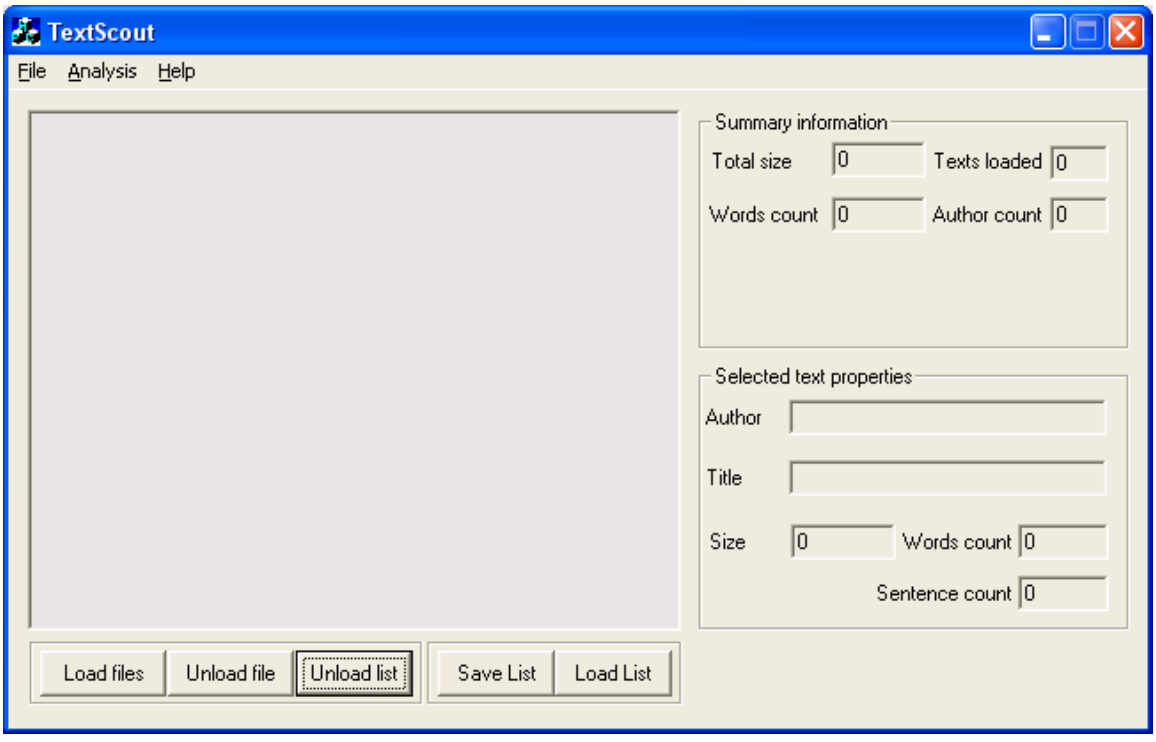

Рис. 3. Общий вид программы

Общий вид программы (см. Рис. **3**3) содержит окно для отображения списка загруженных текстов (слева), суммарную информацию по загруженным текстам (справа и вверху), свойства выбранного в списке текста (справа и внизу), а также главное меню.

Суммарная информация по всем текстам отображает общий размер загруженных файлов, количество загруженных текстов, общее число слов во всех текстах, а также число

различных авторов во всех текстах. В том случае, если предполагается использовать текст в процессе поиска набора характеристик, в нем явным образом (по определенным правилам) задается автор и название произведения. В том случае, если текст не содержит явного указателя авторства, автор считается неизвестным. Все неизвестные авторы считаются как один авторов.

В разделе свойств выбранного текста отображается автор текста, если он был указан явным образом, название, размер, число слов и предложений в тексте.

Под окном со списком загруженных текстов находятся кнопки управления загрузкой, о назначении которых будет сказано в следующем разделе. Главное меню содержит три пункта: File (для управления загрузкой текстов), Analysis (все функции анализа текста) и Help (контекстная помощь).

5.6.2. Загрузка текстов

Для того чтобы произвести анализ текста, либо включить его в процесс поиска набора характеристик, программе нужно целиком загрузить его в оперативную память. Это необходимо, так как в ходе генетического поиска происходит многократное обращение к текстам, что обрабатывается гораздо быстрее, когда все тексты имеются в памяти (безусловно, машине должно хватать ресурсов, чтобы произвести загрузку). Для удобства обработки текст загружается как двунаправленный список слов. Таким образом, лексический анализатор, производящий расчленение текст на лексемы работает единственный раз при загрузке.

Рассмотрим более подробно кнопки, расположенные под окном загрузки (полный их эквивалент можно найти в главном меню в пункте File):

Load files – загрузить один или более файлов из одной директории;

Unload file – выгрузить файл из памяти и исключить из списка;

Unload list – выгрузить все файлы из памяти;

Save list – записать сформированный список файлов для загрузки его в другой раз;

Load list – загрузить ранее сохраненный список файлов (при этом происходит загрузка каждого файла из списка).

5.6.3. Подсчет наиболее часто встречающихся слов и буквосочетаний

Функции подсчета часто встречающихся слов и буквосочетаний можно найти в меню Analysis (анализ), они используются для поиска материала макроатомов.

Реализация этих функций была развитием мыслей о том, что идея Фоменко с подсчетом служебных слов не лишена смысла. Единственный ее недостаток заключается в том, что служебные слова должны быть заранее определены. Так как желательно находить

характеристики независимо от языка, необходимо было разработать механизм, формирующий список служебных слов автоматически. Подобным механизмом оказался поиск наиболее часто встречающихся слов. Если провести такой поиск для большого числа текстов, то в число первой сотни слов как раз и войдут все нужные нам слова. Конечно, это список будет не полным, и будет содержать далеко не только нужные нам служебные слова (предлоги, союзы и частицы), а также, например, междометия и местоимения. Но вполне вероятно, что и они могут пригодиться в ходе генетического поиска характеристик.

Функция «Top frequency words» формирует нужный список наиболее часто встречающихся слов. Диалог, содержащий настройку поиска и окно вывода, изображен на Рис. **4** 4. В число параметров функции входит лишь количество выдаваемых часто встречающихся слов. После осуществления поиска список можно сохранить в файле, и в дальнейшем использовать слова из этого списка в качестве макроатомов генетического алгоритма. Поиск в таком случае будет осуществляться с использованием этих слов целиком в предположении, что эти слова обладают особенными свойствами. Теоретически подобная операция необходима лишь в качестве ускорения, так как, если эти слова на самом деле важны, они будут сформированы и без загрузки их как макроатомов. Список первых пятидесяти часто встречающихся слов, подсчитанных для того же набора текстов, который использовался нами на протяжении всей работы, дан в Приложении 4.

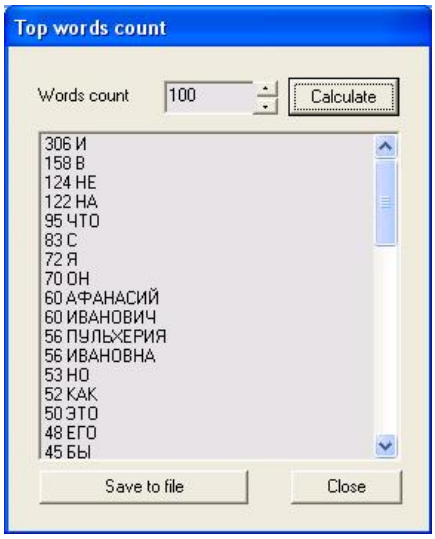

Рис. 4. Подсчет наиболее часто встречающихся слов

Функция «Top frequency n-grams» (поиск часто встречающихся буквосочетаний) аналогична функции поиска часто встречающихся слов. Только в данном случае рассматриваются не просто слова, а буквосочетания определенной длины, которую можно задать предварительно, в каждом слове. Функция написана в предположении, что особыми свойствами обладают не только служебные слова, но и часто встречающиеся буквосочетания.

5.6.4. Сбор статистики

Функция сбора статистики (Gather statistics) расположена в меню «Analysis», и предназначена для сбора статистики по предварительно заданному шаблону (варианту, генетической строке). Диалог сбора статистики (см. **Ошибка! Источник ссылки не найден.**5) позволяет выбрать файл, содержащий генотип (шаблон для сбора статистики),

файл для записи результатов, размер выборки, а также загрузить макроатомы из файла. После выполнения функции файл результатов будет содержать матрицу векторов-столбцов, которую можно подвергнуть дальнейшей обработке и использовать в «Визуализации данных».

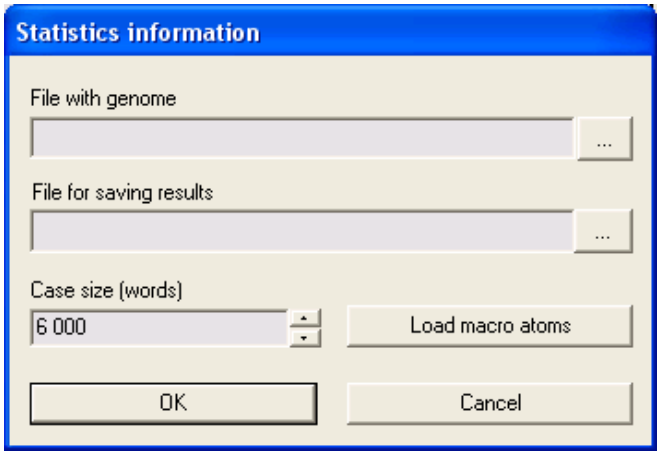

Рис. 5. Сбор статистической информации

5.6.5. Визузализация данных

Функция визуализации данных, расположенная в меню «Analysis», позволяет работать с ранее сохраненными матрицами векторов-столбцов статистики, собранными на основе заданного шаблона (генотипа). Диалог визуализации данных (см. Рис. **6**6) предоставляет интерфейс для выбора файла данных, а также метода визуализации. Алгоритмы визуализации данных (сейчас это только различные варианты метода главных компонент) были взяты нами из программной реализации Ф.Муртага (F.Murtagh), найденной в Интернет. Программа была модифицирована (добавлен соответствующий класс) и интегрирована в текст нашей основной программы – TextScout. Метод главных компонент, реализованный в найденной программе, содержал три свои разновидности, каждую из которых можно выбрать в диалоге визуализации данных.

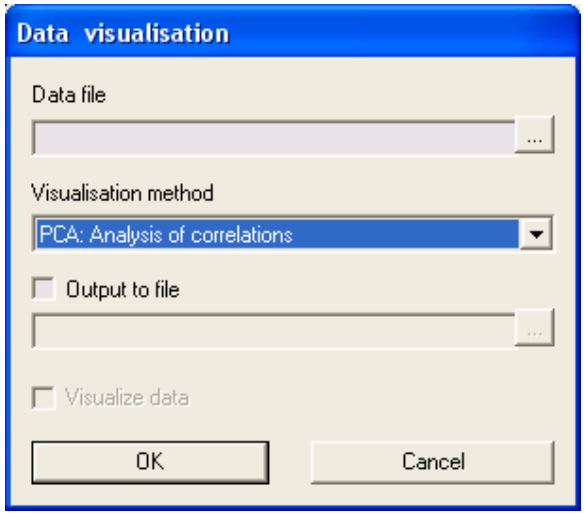

Рис. 6. Визуализация данных

На данный момент визуализация как таковая не была реализована. Функционирует лишь запись редуцированного пространства в файл. Сохраненные данные могут быть отображены и проанализированы в специализированных математических программах (MatLab, Statistica).

5.6.6. Генетический поиск набора характеристик

Функция генетического поиска набора характеристик, как и другие функции анализа, расположена в меню «Analysis». Она осуществляет поиск набора характеристик, мониторинг за процессом поиска, запись промежуточных результатов, а также ведение журнала популяций, которые возникают в ходе поиска.

Параметры генетического поиска задаются в соответствующем диалоге (см. **Ошибка! Источник ссылки не найден.**7). Следует остановиться на каждом из них подробно.

Параметр «First population» (первая популяция) имеет два возможных значения: «get from file» (взять из файла) и «initialize random» (произвести случайную инициализацию). В первом случае первая популяция загружается из файла. В ходе работы алгоритм может сохранять в файле так называемые «restore points» (точки восстановления), и данный режим позволяет продолжить поиск дальше. В случае выбора случайной инициализаций, которая используется при первой попытке поиска, члены популяции определяются случайным образом с помощью функции инициализации, заложенной в программе.

Опция «Save restore points» (сохранять точки восстановления) с возможностью указания файла включает режим записи точек восстановления, о которых говорилось выше.

«Log populations» (записывать журнал популяций) с возможностью выбора файла для записи популяций позволяет осуществлять запись всех популяций и всех их членов, которые возникают в ходе поиска. Данный режим позволяет отслеживать направления генетического поиска, что становится немаловажным, когда поиск заходит в тупик. В этом случае стоит провести реинициализацию и начать поиск с начала или с одного из удачных сохранных моментов.

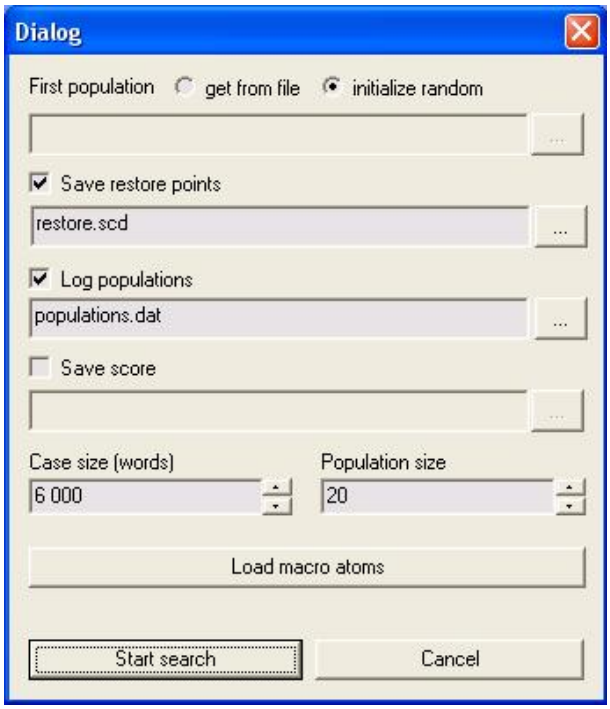

Рис. 7. Генетический поиск набора характеристик

Опция «Save score» (запись счета) позволяет сохранять статистику, предоставляемую библиотекой GALib. В нее входит большинство значимых переменных генетического алгоритма.

Помимо различных функций сохранения статистики диалог настройки генетического поиска предоставляет выбор размера выборки, размера популяции, а также позволяет сделать загрузку макроатомов, которые будут участвовать в генетическом поиске.

Нажатие кнопки «Start search» повлечет за собой запуск генетического алгоритма и вызов нового диалога мониторинга процесса поиска (см. **Ошибка! Источник ссылки не найден.**8). Новый диалог содержит информацию о начале запуска (предполагается, что подсчет может длится несколько дней, а может и недель), общем времени работы, начальном значении фитнес-функции лучшего индивидуума, количестве поколений, максимально

достигнутом на данный момент значении фитнес-функции, числе обращений фитнес функции, количестве применений операторов мутации и кроссовингера. Помимо этого в диалоге отображается лучший индивидуум на данный момент поиска, а также информация о кэшировании характеристик – число характеристик в КЭШе, число удаленных характеристик, число характеристик, к которым производилось обращение, число нулевых характеристик, которые извлекались из КЭШа. Процесс поиска может быть в любой момент остановлен и, если была включена опция записи точек восстановления, продолжен с сохраненного места.

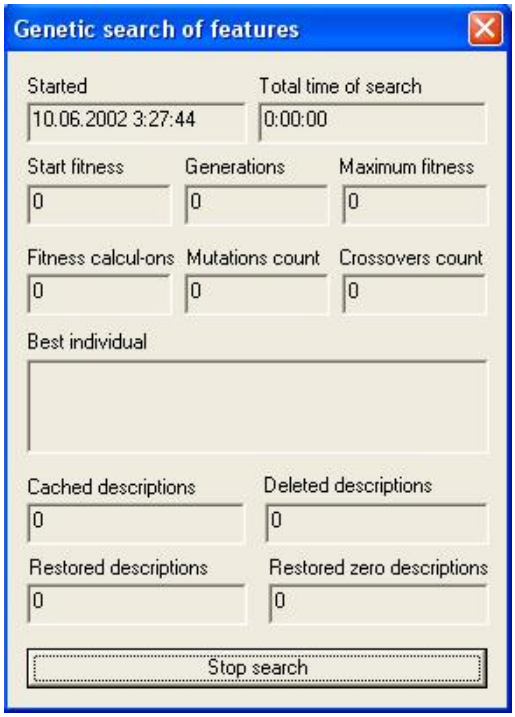

Рис. 8. Мониторинг процесса поиска

5.8. Полученные результаты. Перспективы улучшения и расширения функциональности метода

В ходе тестирования метода нами были рассмотрены различные группы и объемы текстов. Универсальные характеристики, позволяющие разделять любых авторов, к моменту написания работы еще не были найдены, так как процесс поиска носит глобальный характер и работает достаточно долго, но промежуточные результаты имеются уже сейчас. Например, стилевое пространство для семи текстов трех писателей (А.Беляев, И.Ильф и Е.Петров, А.Солженицын) и найденного набора характеристик выглядит следующим образом (см. Рис. 9):

Кластеры выделяются еще недостаточно четко, но уже сейчас заметна тенденция к их формированию. Нами проводились также более масштабные эксперименты и эксперименты на другом наборе текстов, все они показали результаты, подтверждающие то, что метод эффективен, но нужно производить его дальнейшее улучшение.

Помимо уже названных выше преимуществ и достоинств метода кластеризации текстов на основе генетического поиска набора характеристик, он обладает потенциалом развития, отдельные моменты которого могут быть реализованы уже сейчас. Хотелось бы остановить внимание на некоторых, наиболее перспективных, направлениях.

В ходе разбора целей генетического поиска набора характеристик, представленного выше, говорилось лишь о нахождении уникального набора характеристик, работающего для любых авторов. Но представленный метод также можно настроить и на поиск уникальных

характеристик одного автора (задача более простого характера), то есть составление слепка (отпечатка) авторского стиля. Для этого достаточно лишь переписать часть фитнес-функции, отвечающей за вычисление единственного значения целевой функции из матрицы векторов-стоблцов. Полученный слепок авторского стиля можно представить в виде описания, что будет бесценной информацией для филологов, занимающихся исследованием стилей писателей.

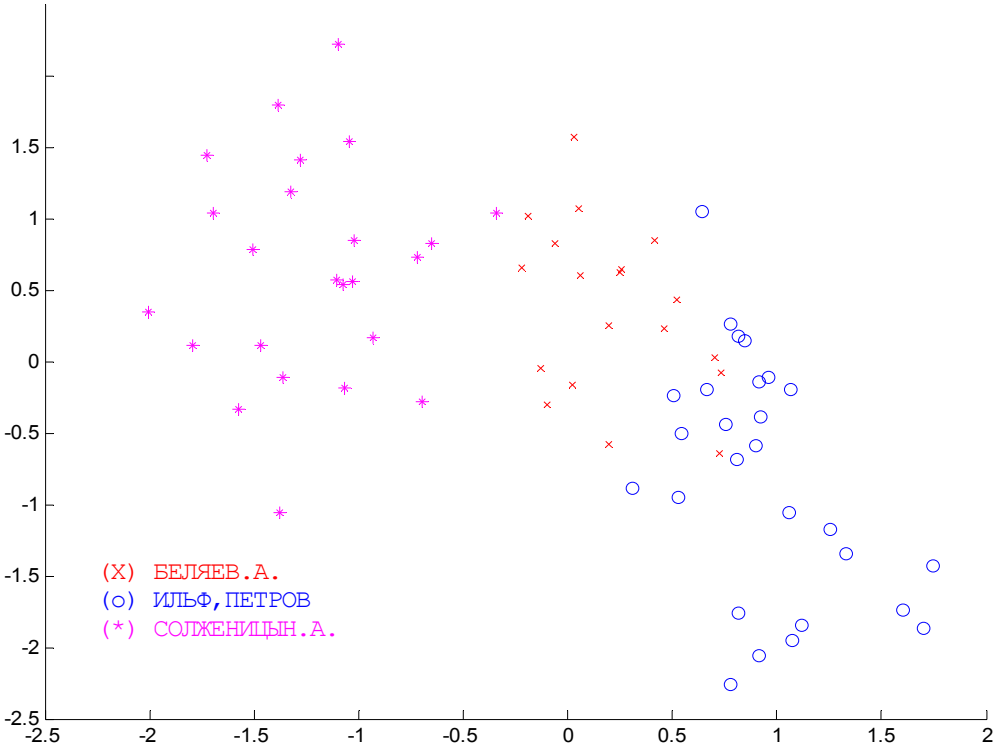

Рис. 9 Стилевое пространство для найденного набора характеристик

Еще одна сфера применения метода заключается в осуществлении стилевой дифференциации, и оценке качества литературного стиля. Вместо того чтобы «отодвигать» только группы отдельных писателей в ходе генетического поиска, можно работать с произвольно сформированными группами. Так, например, можно сделать две группы: одну, состоящую из текстов писателей классиков, другую – из текстов школьных сочинений. Генетический поиск, направленный на разделение этих групп, позволит найти набор характеристик оценивающих качество или «литературность» стиля произведения. Таким же способом можно «научиться» распознавать писателей разных веков или стран, переводные и непереводные тексты и так далее.

Что касается самого процесса генетического поиска, то есть явная возможность улучшения – его визуализация. Помимо общей статистики нужно отображать сами классы, которые программа пытается разделить. В таком случае будет проще осуществлять контроль над процессом поиска и следить, как быстро происходят качественные скачки популяций.

ЗАКЛЮЧЕНИЕ

Полученные в ходе исследования результаты дают основания полагать, что бессознательные характеристики авторского стиля существуют, и методы искусственного интеллекта обладают необходимым потенциалом для их нахождения. Нейронные сети и генетические алгоритмы без труда конкурируют с уже имеющимися методами стилеметрии благодаря прозрачности своего использования и простоте реализации конечных программ определения стиля.

Подход, основанный на фиксированной оценке принадлежности стиля, реализованный в данной работе, успешно решает задачу распознавания авторов, но обладает рядом принципиальных недостатков, затрудняющих его практическое использование. Эти недостатки были преодолены в принципиально ином методе кластеризации текстов на базе поиска характеристик с помощью генетических алгоритмов. Данный метод, предложенный и реализованный нами в работе, основывается на идее автоматического нахождения универсального набора характеристик, позволяющего формировать наглядные отображения стилевой карты текстов. Поиск осуществляется с помощью генетических алгоритмов и, несмотря на все его недостатки, касающиеся трудоемкости и ресурсоемкости, обладает большим потенциалом разрешения самых различных проблем, связанных с анализом авторских стилей.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Д. Хмелёв. Краткая история разработки методик определения авторского стиля. (http://www.rusf.ru/books/analysis/history.htm)
2. Головин Б.Н. Язык и статистика. – М.: Изд-во «Просвещение», 1971. – 189с.
3. Морозов Н.А. Лингвистические спектры: средство для отличения плагиатов от истинных произведений того или иного неизвестного автора. Стилеметрический этюд. // Известия отд. русского языка и словесности Имп.Акад.наук, Т.XX, кн.4, 1915.
4. Марков А.А. Об одном применении статистического метода. // Известия Имп.Акад.наук, серия VI, Т.X, N4, 1916. – с.239.
5. Фоменко В.П., Фоменко Т.Г. Авторский инвариант русских литературных текстов. Предисловие А.Т. Фоменко. // Фоменко А.Т. Новая хронология Греции: Античность в средневековье. Т. 2, – М.: Изд-во МГУ, 1996. – с.768-820.
6. Тютерев В.В., Шевелев О.Г., Использование нейронных сетей GTM для редукции многомерных пространств // VI межвузовская конференция студентов аспирантов и молодых ученых "Наука и образование", – Томск: ТГПУ, 2001, с. 97-99.
7. Farringdon Jill M. Analyzing for Authorship: A Guide to the Cusum Technique, – Cardiff: University of Wales Press, 1996.
8. Morton A.Q. The Authorship of Greek Prose // Journal of the Royal Statistical Society (A), 128, 1965. – p. 169-233.
9. Ashford T. Computerised Determination of Disputed Authorship: The Cusum Method, 2001. – 61p.
10. Hersee M.S. Automatic Detection of Plagiarism: An approach Using the Qsum Method, – University of Sheffield, Department of Computer Science, 2001. – 67p.
11. Hardcastle R.A. CUSUM: a credible method for the determination of authorship? // Science & Justice 37, 1997. – p. 129-138.
12. Tweedie F.J., Singh S., Holmes D.I. Neural Network Applications in Stylometry: The Federalist Paper // Computers and the Humanities 30, 1996. – p. 1-10.
13. Lowe D., Matthews R. Shakespeare vs. Fletcher: A Stylometric Analysis by Radial Basis Functions // Computers and the Humanities 29, 1995. – p. 449-461.
14. Библиотека Максима Мошкова. (http://lib.ru)
15. Тютерев В.В., Шевелев О.Г., Анализ текстов с помощью семантических карт // Нейроинформатика и ее приложения: материалы IX Всероссийского семинара, Под общей ред. А.Н.Горбаня, 2001, Красноярск: ИПЦ КГТУ, С. 199-200.

16. Goldberg D.E. Genetic algorithms in search, optimization and machine learning. Addison-Wesley, Reading, 1989.
17. Holland J.H. Adaptation in Natural and Artificial Systems. Univ. of Michigan Press., Second Ed.1992, MA: The MIT Press edition, 1975.
18. Исаев С. Генетические алгоритмы – эволюционные методы поиска
19. Змитрович А. И. Интеллектуальные информационные системы. – Минск: ТетраСистемс, 1997. – 367с.
20. Главные компоненты и факторный анализ/Учебник StatSoft, – 1984-1998.
21. Honkela T., Pulkki V., Kohonen T., Contextual Relations of Words in Grimm Tales, Analyzed by Self-Organizing Map, – Proceedings of International Conference on Artificial Neural Networks, 1995.
22. Platt J.C., AutoAlbum: Clustering Digital Photographs using Probabilistic Model Merging, – Proc. IEEE Workshop on Content-Based Access of Image and Video Libraries, 2000.
23. Тютерев В.В., Новосельцев В.Б., Теория нейронных сетей, – Томск: Изд. Томского государственного университета, 2000. – 56с.
24. Matthew W., GAlib: A C++ Library of Genetic Algorithm Components, version 2.4, Documentation Revision B, – Massachusetts Institute of Technology, August 1996. – 104p.

ПРИЛОЖЕНИЕ 1. СЛУЖЕБНЫЕ СЛОВА, ВЗЯТЫЕ ФОМЕНКО В КАЧЕСТВЕ АВТОРСКОГО ИНВАРИАНТА

ПРЕДЛОГИ:

в	на	с	за	к
по	из	у	от	для
во	без	до	о	через
со	при	про	об	ко
над	из-за	из-под	под	

СОЮЗЫ:

и	что	но	а	да
хотя	когда	чтобы	если	тоже
или	то есть	зато	будто	

ЧАСТИЦЫ:

не	как	же	даже	бы
ли	только	вот	то	ни
лишь	ведь	вон	то-есть	нибудь
нибудь	уже	либо		

Итого: 55 служебных слов. Хотя список неполон, в случае Фоменко он оказался вполне достаточным для различения авторов.

ПРИЛОЖЕНИЕ 2. ПЕРЕЧЕНЬ АВТОРОВ И ПРОИЗВЕДЕНИЙ, ИСПОЛЬЗУЕМЫХ ДЛЯ ОБУЧЕНИЯ АЛГОРИТМОВ АВТОМАТИЧЕСКОГО УСТАНОВЛЕНИЯ АВТОРСТВА

1. А.Беляев:
   1.1. «Голова профессора Доуэля»;
   1.2. «Остров погибших кораблей»;
   1.3. «Человек нашедший свое лицо»;
   1.4. «Звезда Кэц»;
2. Н.В.Гоголь:
   2.1. «Вечера на хуторе близ Диканьки» ч. I,II;
   2.2. «Тарас Бульба»;
   2.3. «Повесть о том, как поссорился Иван Иванович с Иваном Никифоровичем»;
   2.4. «Старосветские помещики»;
   2.5. «Вий»;
3. И.А.Гончаров:
   3.1. «Обломов»;
   3.2. «Обыкновенная история»;
4. Ф.М.Достоевский:
   4.1. «Бедные люди»;
   4.2. «Бесы»;
   4.3. «Идиот»;
   4.4. «Братья Карамазовы»;
   4.5. «Преступление и наказание»;
5. И.Ильф и Е.Петров:
   5.1. «Двенадцать стульев»;
   5.2. «Золотой теленок»;
   5.3. «Одноэтажная Америка»;
6. В.Набоков:
   6.1. «Лолита»;
   6.2. «Камера обскура»;
   6.3. «Защита Лужина»;
   6.4. «Другие берега»;
   6.5. «Подвиг»;
7. А.С.Пушкин:

7.1. «Капитанская дочка»;

7.2. «Дубровский»;

7.3. «Повести покойного Ивана Петровича Белкина»;

8. А.Солженицын:

8.1. «Один день Ивана Денисовича»;

8.2. «Архипелаг ГУЛаг Том 1»;

8.3. «Матренин двор»;

8.4. «Раковый корпус»;

9. Л.Н.Толстой:

9.1. «Анна Каренина»;

9.2. «Хаджи-Мурат»;

9.3. «Детство»;

9.4. «Отрочество»;

9.5. «Юность»;

10. С.Грин:

10.1. «Алые паруса»;

10.2. «Бегущая по волнам»;

10.3. «Блистающий мир»;

10.4. «Золотая цепь».

Всего: 10 писателей и 40 текстов общим объемом 21 Мб.

ПРИЛОЖЕНИЕ 3. РЕЗУЛЬТАТЫ РАБОТЫ ПРОГРАММЫ AUS

Н.С.Лесков
Старые годы в селе Плодомасове

A.Belyaev:	0.000
N.V.Gogol:	0.051
I.A.Goncharov:	0.041
F.M.Dostoevsky:	0.000
Ilf & Petrov:	0.000
V.Nabokov:	0.058
A.S.Pushkin:	0.038
A.Solzhenycyn:	0.330
L.Tolstoy:	0.311
A.S.Grin:	0.013

Н.С.Лесков
Загадочный человек

A.Belyaev:	0.079
N.V.Gogol:	0.000
I.A.Goncharov:	0.000
F.M.Dostoevsky:	0.000
Ilf & Petrov:	0.000
V.Nabokov:	0.204
A.S.Pushkin:	0.051
A.Solzhenycyn:	0.485
L.Tolstoy:	0.373
A.S.Grin:	0.000

Л.Н.Толстой
Воскресение

A.Belyaev:	0.024
N.V.Gogol:	0.000
I.A.Goncharov:	0.000
F.M.Dostoevsky:	0.000
Ilf & Petrov:	0.008
V.Nabokov:	0.039
A.S.Pushkin:	0.000
A.Solzhenycyn:	0.000
L.Tolstoy:	1.000
A.S.Grin:	0.000

Л.Н.Толстой
Смерть Ивана Ильича

A.Belyaev:	0.044
N.V.Gogol:	0.000
I.A.Goncharov:	0.040
F.M.Dostoevsky:	0.000
Ilf & Petrov:	0.006
V.Nabokov:	0.009
A.S.Pushkin:	0.023
A.Solzhenycyn:	0.005
L.Tolstoy:	1.000
A.S.Grin:	0.000

А.Грин
Сердце пустыни

A.Belyaev:	0.046
N.V.Gogol:	0.000
I.A.Goncharov:	0.000
F.M.Dostoevsky:	0.016
Ilf & Petrov:	0.005
V.Nabokov:	0.139
A.S.Pushkin:	0.000
A.Solzhenycyn:	0.077
L.Tolstoy:	0.419
A.S.Grin:	0.650

А.С.Грин
Повесть, оконченная благодаря пуле

A.Belyaev:	0.294
N.V.Gogol:	0.000
I.A.Goncharov:	0.000
F.M.Dostoevsky:	0.112
Ilf & Petrov:	0.089
V.Nabokov:	0.008
A.S.Pushkin:	0.053
A.Solzhenycyn:	0.000
L.Tolstoy:	0.000
A.S.Grin:	0.570

Ф.М.Достоевский
Двойник

A.Belyaev:	0.071
N.V.Gogol:	0.007
I.A.Goncharov:	0.039
F.M.Dostoevsky:	0.986
Ilf & Petrov:	0.000
V.Nabokov:	0.064
A.S.Pushkin:	0.000
A.Solzhenycyn:	0.000
L.Tolstoy:	0.008
A.S.Grin:	0.000

Ф.М.Достоевский
Село Степанчиково и его обитатели

A.Belyaev:	0.061
N.V.Gogol:	0.000
I.A.Goncharov:	0.113
F.M.Dostoevsky:	1.000
Ilf & Petrov:	0.000
V.Nabokov:	0.002
A.S.Pushkin:	0.006
A.Solzhenycyn:	0.000
L.Tolstoy:	0.000
A.S.Grin:	0.009

И.С.Тургенев
Ася

A.Belyaev:	0.063
N.V.Gogol:	0.071
I.A.Goncharov:	0.068
F.M.Dostoevsky:	0.064
Ilf & Petrov:	0.000
V.Nabokov:	0.031
A.S.Pushkin:	0.328
A.Solzhenycyn:	0.000
L.Tolstoy:	0.137
A.S.Grin:	0.000

В.Тютерев, О.Шевелев
Нейронные сети в задаче определения авторства текста

A.Belyaev:	0.000
N.V.Gogol:	0.000
I.A.Goncharov:	0.000
F.M.Dostoevsky:	0.060
Ilf & Petrov:	0.291
V.Nabokov:	0.638
A.S.Pushkin:	0.118
A.Solzhenycyn:	0.193
L.Tolstoy:	0.041
A.S.Grin:	0.014

ПРИЛОЖЕНИЕ 4. СПИСОК САМЫХ ЧАСТО ВСТРЕЧАЮЩИХСЯ СЛОВ ДЛЯ 10 ВЫБРАННЫХ АВТОРОВ И ИХ 40 ПРОИЗВЕДЕНИЙ

129345 И
72655 В
65286 НЕ
46786 ЧТО
44417 НА
37613 Я
37575 ОН
35459 С
25772 КАК
23688 А
21146 НО
19929 ЕГО
19273 ЭТО
17192 К
17117 ОНА
15683 ВСЕ
15098 ТАК
14343 ЖЕ
13211 БЫЛО
13142 ЗА
13109 ПО
13012 У
12160 ВЫ
11363 БЫ
10664 ИЗ
10135 ТОЛЬКО
10100 ОТ
10085 ТО
10083 ЕЕ
9849 О
9500 ДА
9489 МЕНЯ
9292 МНЕ
9182 ЕЩЕ
8746 БЫЛ
8056 УЖЕ
8032 ТЫ
7941 ЕМУ
7112 ВОТ
7108 СКАЗАЛ
6926 КОГДА
6585 МЫ
6385 ТЕПЕРЬ
6230 ЕСЛИ
6004 ДАЖЕ
5938 НИ
5888 ДЛЯ
5814 ИЛИ
5790 БЫТЬ
5774 НЕТ

ПРИЛОЖЕНИЕ 5. РУКОВОДСТВО ПРОГРАММИСТА

В нашей работе было реализовано два метода: установление авторства с фиксированным набором альтернатив (нейронные сети прямого распространения) и кластеризация текстов с помощью универсального набора характеристик (генетические алгоритмы и редукция многомерных пространств). Первый метод демонстрационный, выступающий в качестве промежуточной ступени, не предполагающий дальнейшего развития, и поэтому нет смысла раскрывать его внутреннюю программную структуру. Что касается реализации метода кластеризации текстов, то для работы с ним (улучшения и развития программы) нужно четко представлять весь набор классов, структуру данных и множество используемых библиотек.

В последней версии программы присутствуют 16 классов данных (использовалась объектно-ориентированная парадигма разработки), не считая классов библиотеки GALib. Среди них 8 классов диалогов (форм), 1 приложения и 7 концептуальных. Для понимания взаимосвязей модулей программы, прежде всего, нужно разобрать классы диалогов (класс приложения присутствует в каждой программе и не несет смысловой нагрузки). Ход разъяснений рассчитан на то, что программист уже знаком с внешним представлением программы и знает ее на уровне руководства пользователя.

CAboutDlg – класс окна вывода информации о программе.

CTextScoutDlg – основной диалог программы (см. Общий вид программы, Руководство пользователя). Содержит методы запуска различных функций, отвечает за загрузку текстов в память, считает суммарную информацию о загруженных файлах. Хранит методы для работы со списками загружаемых текстов и удалением файлов из списка. Наиболее значимая структура данных *m_cTexts*, представляющая собой список указателей на тексты (**CLitText**). Отображаемые на экран названия синхронизируются с этой структурой и содержатся в *m_cFiles* (MFC тип **CListBox**). Диалог также содержит указатели на окно отображения прогресса загрузки (**CProgressDlg**) и поток, в котором происходит загрузка (**CWinThread**). Функция загрузки (**LoadThreadFunc**), выполняющаяся в отдельном потоке, не принадлежит классу, но располагается в том же файле.

CProgressDlg – диалог отображения прогресса состояния. Вызывается главным окном программы (**CTextScoutDlg**) для отображения прогресса загрузки, а также функциями подсчета наиболее встречающихся слов и буквосочетаний (**CTopObjectsDlg**). Содержит единственный метод-событие, срабатывающее по нажатию кнопки Cancel в диалоге. Отправляет сообщение о прерывании процесса, вызвавшему его окну.

CTopObjectsDlg – диалог, управляющий поиском наиболее часто встречающихся слов и буквосочетаний. Его вид зависит от параметров инициализации – либо дает настроить поиск часто встречающихся слов, либо буквосочетаний. Наиболее значимая и специфическая структура данных – кэш различных слов (*CMapStringToOb wordHash*), с помощью которого осуществляется сбор уникальных объектов (слов или буквосочетаний) и подсчет количества их повторений. С началом поиска вызывает **CProgressDlg** для отображения процесса обработки.

CStatDlg – диалог функции сбора статистки (Gather statistics). Позволяет выбрать файл с генотипом-шаблоном сбора статистики, файл для результатов, а также размер выборки. Сбор статистики осуществляется с помощью класса **CStat**, единственная значимая функция-событие – *OnGenomeFile()*, загружающая генотип из файла.

CVisDataDlg – диалог функции визуализации данных. Обладает лишь интерфейсными методами. Работает вместе с классами **CStat** и **CPCA_routines**. Основная значимая структура данных – *data* (указатель на двумерный массив, куда загружаются данные).

CGeneticDlg – основной диалог, осуществляющий управление запуском генетического поиска. Обладает объектами настройки параметров поиска, подготавливает класс генотипа и генетического алгоритма к работе. Устанавливает параметры по умолчанию. Содержит ссылки на класс сбора статистики **CStat**, поток, в котором происходит поиск, и диалог отображения прогресса генетического поиска (**CGenProgDlg**). Располагает методами управлениями процесса поиска, обновления отображения состояния, записи журнала поиска и слежением за временем работы алгоритма. Файл класса содержит также функцию, осуществляющую поиск и запускающуюся в отдельном потоке, переопределенные методы инициализации, мутации, кроссовингера и фитнес-функцию генетического алгоритма. Содержит функции для кэширования характеристик (**CCachedDesc**), проверки числа обращений к ним, модификации генетической строки для того, чтобы кэшированные характеристики не подсчитывались дважды, и их удаления.

CGenProgDlg – диалог отображения прогресса генетического поиска. Содержит множество полей, сообщающих состояние процесса поиска (см. Общий вид программы, Руководство пользователя) и функцию прерывания процесса поиска.

Рассмотрим концептуальные классы программы:

Центральными классами, которые используют все функции программы, являются **CLitLexeme** и **CLitText**.

CLitLexeme – класс «литературная лексема». В ходе работы лексического анализатора выделяются отдельные блоки текста (как правило, слова), которые имеют общее название –

лексемы. Данный класс содержит все необходимые атрибуты подобных блоков и методы установления их типа.

CLitText – класс «литературный текст». Один из наиважнейших классов программы. Центральная структура данных – список слов текста (*m_cWords*). Класс также содержит информацию об авторе, названии, количестве слов, предложений, количестве различных слов, исходном файле произведения. Позволяет считывать текст в список слов, загружать информацию о нем, собирать статистику по заданной генетической строке и скапливать информацию в переданной по ссылке структуре. Класс «литературный текст» содержит встроенный лексический анализатор (с таблицей анализа), обеспечивающий необходимое разбиение текста на лексемы в процессе загрузке.

CPCA_routines – класс, образованный на основе программы Ф.Муртага (F.Murtagh, Munich, 6 June 1989), производящей редукцию многомерных признаковых пространств методом главных компонент (3 модификации). В нашей программе используется для приведения многомерных пространств к двумерным для последующей их визуализации. Является способом представления результатов в доступном для человека виде.

CStat – класс, организующий работу с генетическими строками, как носителями информации о собираемой статистике по текстам. Содержит функции, необходимые для работы со списком атомов (базовым и рабочим), формирования указателей на расположение отдельных характеристик и проверки их корректности. Также включает в себя блок подсчета значения фитнес-функции на основе матрицы векторов-столбцов характеристик по принципу отношения дисперсии классов к дисперсии внутри классов писателей.

CAuthorGroup – вспомогательный класс «группа произведений автора». Содержит поля: авторское имя, дисперсия внутри группы, центр группы, индекс начала и конца произведений данного автора в списке всех произведений (произведения одного автора в списке обязательно должны идти один за другим, перемешивание разных произведений не допускается).

CCachedDesc – вспомогательный класс «кэшированная характеристика». Хранит информацию о поколении, на котором была выработана данная характеристика, генетической строке, которая ее представляет, числе обращений к характеристике и поколении, на котором произошло последнее обращение.

В ходе работы с библиотекой GALib также были задействованы следующие классы:

GASimpleGA

GA1DArrayGenome<int>

GANoScaling

Подробности их использования можно посмотреть в [24].