Г. В. Дорохина, Пофонемное распознавание как задача классификации рядов на множестве последовательностей элементов сложных объектов с применением усовершенствованного trie-дерева, *Информатика и автоматизация*, 2024, выпуск 23, том 6, 1784–1822

DOI: 10.15622/ia.23.6.8

УДК 004.89:004.04 **DOI** 10.15622/ia.23.6.8

Г.В. Дорохина

ПОФОНЕМНОЕ РАСПОЗНАВАНИЕ КАК ЗАДАЧА КЛАССИФИКАЦИИ РЯДОВ НА МНОЖЕСТВЕ ПОСЛЕДОВАТЕЛЬНОСТЕЙ ЭЛЕМЕНТОВ СЛОЖНЫХ ОБЪЕКТОВ С ПРИМЕНЕНИЕМ УСОВЕРШЕНСТВОВАННОГО TRIE-ДЕРЕВА

Дорохина Г.В. **Пофонемное распознавание как задача классификации рядов на множестве последовательностей элементов сложных объектов с применением усовершенствованного trie-дерева.**

Аннотация. Последовательности, в том числе последовательности векторов, применимы в любых предметных областях. Последовательности скалярных значений или векторов (ряды) могут быть порождены последовательностями более высокого порядка, например: последовательностями состояний, элементов сложных объектов. Работа посвящена применению усовершенствованного trie-дерева в задаче классификации ряда на множестве последовательностей элементов сложных объектов методом динамического программирования. Рассмотрены сферы применения динамического программирования. Показано, что динамическое программирование приспособлено к многошаговым операциям вычисления аддитивных (мультипликативных) мер подобия / различия. Утверждается, что усовершенствованное trie-дерево применимо в задаче классификации ряда на множестве последовательностей элементов сложных объектов методом динамического программирования при использовании таких мер подобия / различия. Выполнен анализ иерархических представлений множеств последовательностей. Описаны преимущества, которые обеспечивает усовершенствованное trie-дерево по сравнению с традиционными представлениями других сильноветвящихся деревьев. Разработано формальное описание усовершенствованного trie-дерева. Дано пояснение ранее полученным данным о существенном приросте скорости операций добавления и удаления последовательностей в усовершенствованном trie-дереве относительно использования массива с индексной таблицей (24 и 380 раз, соответственно). Выполнена постановка задачи пофонемного распознавания речевых команд как задачи классификации ряда на множестве последовательностей элементов сложных объектов и изложен метод её решения. Разработан метод классификации ряда на множестве последовательностей элементов сложных объектов с применением усовершенствованного trie-дерева. Он исследован на примере пофонемного распознавания с иерархическим представлением словаря классов речевых команд. В этом методе распознавание речевых команд выполняют в процессе обхода усовершенствованного trie-дерева, хранящего множество транскрипций речевых команд – последовательностей транскрипционных символов, которые обозначают классы звуков. Численные исследования показали, что классификация ряда как последовательности элементов сложных объектов повышает частоту правильной классификации по сравнению с классификацией ряда на множестве рядов, а применение усовершенствованного trie-дерева сокращает затраты времени на классификацию.

Ключевые слова: trie-дерево, множество последовательностей, классификация рядов на множестве последовательностей элементов сложных объектов, динамическое программирование, пофонемное распознавание речевых команд.

1. Введение. Модели структурирования данных для скоростного поиска строк разрабатывают с конца 50-х годов XX века. Систематизировали и развивали методы представления данных А. Ахо и Дж. Ульман, Н. Вирт [1], Д. Кнут [2], Р. Бриандe [3], Д. Гасфилд [4].

Структурой данных, обеспечивающей наибольшую скорость поиска во множестве последовательностей, является дерево цифрового поиска – то же, что лучевой поиск Р. Бриандe [3], trie-дерево, «префиксное дерево», «бор». Особенность организации trie-дерева состоит в том, что элементы последовательности размещены вдоль ветвей дерева. Допустим, задано множество последовательностей символов. Пусть у некоторых из этих последовательностей n начальных символов совпадают со строкой s длины n. В trie-дереве начальные символы указанных последовательностей представлены фрагментом ветви из n вершин. При поиске строки s в указанном множестве её сопоставление с общей начальной частью множества последовательностей происходит однократно. То есть, trie-дерево является перспективной структурой данных для обработки множеств последовательностей, о чем свидетельствует возобновление внимания к этой структуре данных.

Последнее время область применения методов анализа последовательностей значительно расширилась. В виде последовательности представляют информацию о состоянии объектов различной природы. Анализ последовательностей используют в социальных науках [5] и здравоохранении [6, 7], в биологии [4], в экономической сфере [8 – 10], в технике [11 – 13] и т.д.

Многие реальные приложения, такие как веб-майнинг, анализ текста, биоинформатика, системная диагностика и распознавание действий, имеют дело с последовательными данными. В этих приложениях решается задача поиска содержательных шаблонов или создания эффективных прогностических моделей [14], для чего применяют методы машинного обучения, как с учителем, так и без, например, K-средних или метод опорных векторов (SVM). Известное решение в области интеллектуального анализа данных заключается в использовании последовательных шаблонов [15]. Этот подход сначала добывает последовательные шаблоны из набора данных, а затем представляет каждую последовательность в наборе данных как вектор признаков с двоичными компонентами, указывающими, содержит ли эта последовательность конкретный последовательный шаблон. Число последовательных шаблонов часто очень велико. Это

приводит к проблемам с высокой размерностью и разреженностью данных [15].

В некоторых случаях необходимо выполнить нечеткий поиск последовательности во множестве – определить наиболее сходную или отличающуюся последовательность по отношению ко входной. Нечеткий поиск в данном случае можно рассматривать как задачу классификации на множестве последовательностей. Примеры нечеткого поиска в словаре строк: проверка орфографии, распознавание текстов, обнаружение фейковых сайтов [16], поиск нежелательного контента в социальных сетях, поиск дублирующихся данных в базах данных, поиск цитирований и др. Сходство между последовательностями определяют попарно [17, 18]. Для пар строк применяют расстояния Хемминга, Левенштейна, Дамерау-Левенштейна. Обзору мер расхождения между последовательностями посвящена работа [19].

Классические методы сопоставления пар последовательностей разной длины основаны на динамическом программировании. С его помощью последовательности «выравнивают» и сопоставляют, в результате чего определяют их «меру подобия» или «меру различия».

Динамическое программирование относят к перечислительным техникам точных методов поиска решений оптимизационных задач. Это «метод оптимизации, приспособленный к операциям, в которых процесс принятия решений может быть разбит на отдельные этапы (шаги)» [20], – многошаговым операциям. Основоположник метода – Р. Беллман. Методом динамического программирования «могут решаться задачи, приводящиеся к сетевым моделям», например: «транспортные задачи с произвольной … функцией затрат; задачи замены оборудования…; задачи управления запасами и другие» [20].

Динамическое программирование как метод многокритериальной оптимизации [21] заменяет одновременный выбор значений большого числа переменных решаемой экстремальной задачи поочередным определением каждой из них. Выбор значений переменных трактуют как многоэтапный процесс управления дискретной системой, имеющей конечное множество состояний, и управляемой в дискретном времени (пошагово) путём применения воздействий из конечного множества, в результате чего изменяется состояние системы. Определены начальное и множество финальных состояний системы, с изменением состояния системы связан платеж. Последовательности управлений, переводящие систему из начального

в одно из финальных состояний, определяют полные траектории движения системы. Задача состоит в поиске полной траектории, оптимальной по значению суммарного платежа [21].

Динамическое программирование используют как для решения статических задач, например, связанных с распределением ресурсов, так и для задач, связанных с динамикой процесса или системы, применяют для решения задач управления [22]. Упрощение процесса решения достигают за счет ограничения области и количества вариантов, исследуемых при переходе к очередному этапу [22].

Недостатками динамического программирования (ДП) считают следующие проблемы [22]. «Проблема ДП-1» состоит в отсутствии единого универсального метода решения, поскольку каждая задача имеет свои особенности и требует поиска приемлемой совокупности методов решения. «Проблема ДП-2» связана с большими объемами и трудоемкостью решения многошаговых задач, имеющих множество состояний, что приводит к необходимости отбора задач малой размерности либо использования сжатой информации.

Длительное время динамическое программирование применяли в задачах анализа текста и речи, где данные представимы в виде последовательностей разной длины. В задачах обработки символьных последовательностей метод динамического программирования развивали: В.И. Левенштейн, Ф. Дамерау, Р. Вагнер и М. Фишер, Д. Кнут, Д. Моррис и В. Пратт; Р. Бойер и Дж. Мур, А. Ахо и М. Корасик, С. Нидлман и К. Вунш, П. Смит, М. Ватерман; в задачах обработки речи – Т.К. Винцюк [23], Х. Сакое и С. Чиба, Л. Рабинер, В.Ю. Шелепов [24]. Многие методы и алгоритмы, которые зарождались и развивались в связи с задачами распознавания речи, сейчас находят применение в других областях – робототехника, медицина, биоинформатика и др.

Сопоставление и выравнивание пары последовательностей разной длины реализует алгоритм нелинейного растяжения-сжатия временной оси (Dynamic Time Warping, DTW). Его в настоящее время применяют [25 – 29] не только в распознавании речи и строковых алгоритмах, но и в вычислительной геометрии, в различных областях для анализа упорядоченных по времени данных, включая видео, аудиофайлы, временные ряды, измерения и отслеживание данных GPS. Алгоритм DTW используют при классификации и кластеризации последовательностей, например, в задачах распознавания жестов и верификации подписи, в беспроводной связи, в интеллектуальном анализе данных и др. Применительно

к распознаванию речи этот метод используют при недостатке обучающих данных.

Работы автора [30 – 32] посвящены распознаванию речи на основе динамического программирования. К задачам распознавания речи относят: идентификацию дикторов, распознавание изолированных (отдельно произносимых) слов, распознавание речевых команд, распознавание дискретной речи (распознавание речи при пословной диктовке) и выделение ключевых слов в потоке слитной речи, распознавание слитной речи.

Проблема распознавания образов является одной из центральных проблем искусственного интеллекта. При этом «понятие образа можно определить так: объекты, для которых выполняется отношение эквивалентности (одинаковые объекты) или, по крайней мере, отношение толерантности (похожие объекты), в своей совокупности составляет образ» [33]. Проблема распознавания образов «возникла при изучении физиологических свойств мозга», а именно его способности «отвечать на бесконечное множество состояний внешней среды конечным числом реакций» [33]. Распознавание включает в себя обучение опознаванию и опознавание образов. Обучение опознаванию включает выделение классов, выбор пространства признаков образов, разработку классификатора. Опознавание образов состоит в применении классификатора к объекту.

Распознавание изолированных слов является частным случаем задачи распознавания речевых команд. Под речевой командой, с одной стороны, подразумевается произнесение слова или слитное произнесение последовательности слов, которые отделены от остальной речи межфразовыми паузами. С другой стороны, задан перечень слов и последовательностей слов, которые могут быть результатом распознавания. При распознавании оцифрованный звук преобразуют в ряд (последовательность векторов признаков), в результате обработки которого определяют произнесённое слово и последовательность слов. Так как задача сводится к выбору одного из известных классов, то имеем дело с задачей классификации.

Пофонемным называется «распознавание, при котором эталоны слов составляются из общей для всех слов совокупности эталонных элементов, а эталонные элементы интерпретируются как фонемы или части фонем, ответственные за элементарные участки речевого сигнала» [23, 30]. В качестве эталонных элементов могут выступать, например, аллофоны [30, 31] или дифоны [34].

В рамках разработки метода пофонемного распознавания отдельно произносимых слов (2003 г.) на основе trie-дерева автором

создана структура данных для хранения множества строк фонетических транскрипций слов словаря [30]. Пофонемное распознавание выполнялось в процессе обхода этого дерева. Структура данных обеспечивала: 1) быстрый поиск строк; 2) быстрое распознавание слова большого словаря (тысяча слов) за счет однократного сопоставления общих начальных частей транскрипции с соответствующим фрагментом речи; 3) хранение транскрипций в древовидной структуре (возможность получить написание заданной транскрипции из дерева). Возможности древовидной структуры 1) и 3) использованы для хранения больших множеств строк (содержат миллионы строк) и скоростного поиска в них при разработке модуля декларативного морфологического анализа слов русского языка (2004 г.) [35]. В результате этих работ появилось усовершенствованное дерево цифрового поиска [36] – то же, что усовершенствованное trie-дерево. Данная работа является обобщением и системным изложением исследований усовершенствованного trie-дерева и его возможных областей его применения [37 – 39].

Допустим, выделены классы элементов объектов. Назовем образом некоторое обобщённое описание класса в пространстве признаков. Назовём элементы объекта однотипными, если их описания принадлежат одному пространству признаков. В работе [37] предложено ввести алфавит обозначений для классов однотипных элементов. Это позволяет информацию о множестве сложных объектов, представимых как последовательности элементов, хранить и обрабатывать как множество строк.

В работе [37] показано, что trie-дерево [2, 3] – «представление словаря строк для поиска», а усовершенствованное trie-дерево [36] – «представление словаря строк для хранения и поиска». Представление словаря строк для поиска является поисковой структурой, и само множество строк необходимо хранить во внешней по отношению к этой структуре данных памяти. Исследование затрат памяти «представления словаря строк для хранения и поиска» по сравнению с затратами на хранение «представления словаря строк для поиска» одновременно с самим множеством строк выполнено в работе [38].

Работа [39] посвящена, в том числе, точному описанию структур данных усовершенствованного trie-дерева и его исследованию. Опубликованные в ней данные численных исследований скорости выполнения основных операций на множестве словоформ русского языка (около 2 млн. строк) показывают, что у усовершенствованного trie-дерева по сравнению использованием массива с индексной таблицей скорость добавления и удаления строк

возрастает в 24 и 380 раз соответственно. В данной работе этим значениям даны пояснения.

В работе [31] разработан метод пофонемного распознавания речевых команд малого словаря. Некоторые неточности приведенных в ней алгоритмов устранены в данной работе.

В настоящей работе выполнена постановка задачи пофонемного распознавания речевых команд как задачи классификации ряда на множестве последовательностей элементов сложных объектов; разработан метод классификации ряда на множестве последовательностей элементов сложных объектов с применением усовершенствованного trie-дерева на примере пофонемного распознавания с иерархическим представлением словаря классов речевых команд.

2. Актуальность и анализ состояния проблемы. Алгоритм DTW применяют к последовательностям одномерных значений или векторов (назовём их рядами). При этом многие ряды могут быть проявлениями последовательностей более высокого порядка. Так, фрагменту ряда может соответствовать некоторое состояние объекта наблюдения, а ряду – последовательность состояний (определённый сценарий). Или фрагмент ряда может описывать некоторый элемент объекта (звук слова), а ряд – последовательность элементов – сложный объект (слово). Такое рассмотрение рядов особенно актуально в связи с активным развитием направлений извлечения закономерностей и знаний из последовательностей и из процессов [40], работами по классификации и кластеризации последовательностей [41 – 43], в том числе последовательностей сложных объектов.

Когда на множестве последовательностей сложных объектов выбирают наилучшую в соответствии с заданной целевой функцией последовательность, имеем дело с задачей оптимизации. Когда нужно выбрать класс сложных объектов, которому в наибольшей степени принадлежит рассматриваемый объект, решают задачу классификации. Если мера подобия / различия сложного объекта представима в виде суммы или произведения мер расхождения для элементов объекта, то для её вычисления можно применить динамическое программирование.

Актуально рассмотреть усовершенствованное trie-дерево как способ представления множества последовательностей элементов сложных объектов для решения задач классификации на множестве таких последовательностей при использовании аддитивной или мультипликативной меры подобия / различия. Это снижает вычислительную сложность за счет однократного вычисления меры

подобия / различия для одинаковых начальных фрагментов последовательностей, и тем самым решает «проблему ДП-2». Можно предположить аналогичное применение усовершенствованного trie-дерева для задач оптимизации на множестве последовательностей элементов сложных объектов при использовании аддитивной или мультипликативной целевой функции, что может быть предметом последующих работ.

В связи с разнообразием задач, которые решают методом динамического программирования, трудно проанализировать все аспекты отсутствия единого универсального метода решения задач методом динамического программирования («проблема ДП-1»). При этом актуальным является рассмотрение усовершенствованного trie-дерева как универсального способа представления и множества рядов, и множества последовательностей элементов сложных объектов, который обеспечивает:

– скоростной поиск в большом множестве последовательностей, в том числе нечёткий поиск с применением динамического программирования в задачах классификации;

– быструю модификацию иерархической структуры и минимальные затраты памяти на ее хранение.

3. Цель и задачи работы. Целью работы является характеристика усовершенствованного trie-дерева [36] как способа представления множества последовательностей элементов сложных объектов в задаче классификации ряда на этом множестве методом динамического программирования.

В работе поставлены и решены следующие задачи:

– выполнен анализ иерархических представлений множеств последовательностей;

– сформулированы две новые проблемы сильноветвящихся деревьев, которые решает усовершенствованное trie-дерево;

– сформулированы проблемы сопоставления рядов алгоритмом Dynamic Time Warping;

– разработано формальное описание усовершенствованного trie-дерева с учетом решения проблем сильноветвящихся деревьев;

– выполнена постановка задачи пофонемного распознавания речевых команд как задачи классификации ряда на множестве последовательностей элементов сложных объектов с применением алгоритма Dynamic Time Warping и изложен метод её решения;

– разработан метод классификации ряда на множестве последовательностей элементов сложных объектов с применением усовершенствованного trie-дерева на примере пофонемного

распознавания с иерархическим представлением словаря классов речевых команд.

4. Анализ иерархических представлений множеств последовательностей. Иерархические структуры позволяют быстро искать данные, в связи с чем их используют для создания индексов или специальных поисковых структур. Деревья поиска иерархически разбивают пространство поиска на области в соответствии с некоторыми условиями. Например, предикат корневой вершины бинарного дерева поиска разбивает множество ключей на два подмножества: ключи, значение которых меньше хранимого в вершине, и ключи, значение которых, соответственно, больше. Такое разбиение продолжается рекурсивно до тех пор, пока в подмножестве не останется один ключ.

Как представления множеств последовательностей, деревья можно разделить на две группы, исходя из объема данных, которые подвергают анализу в каждой вершине. К первой принадлежат деревья, в которых вся последовательность является ключом и подлежит анализу в каждой вершине. Примерами могут служить бинарное дерево поиска и сильноветвящиеся деревья [1]: (a,b)-дерево, b-дерево, b^+ дерево, R-дерево, k-d-дерево. К этой же группе принадлежат деревья решений, вершина которых содержит определенный предикат. Тогда как, например, в бинарном дереве поиска во всех вершинах применяют одинаковые предикаты «равно ключу в вершине», «меньше (больше) ключа в вершине», в дереве решения каждой вершине назначен собственный предикат.

Ко второй группе принадлежат деревья, в вершинах которых анализируют часть ключа (один или несколько элементов последовательности). При этом элементы последовательности, которые анализируют в некоторой вершине, предшествуют элементам последовательности, которые анализируют в её вершинах-потомках. К этой группе относят trie-дерево, префиксное и суффиксное дерево, дерево из алгоритма Ахо-Корасик.

Деревья второй группы применяют, в основном, для обработки строк и последовательностей. В то же время перспективным представляется их использование и для других классов последовательностей, например последовательностей образов (классов объектов) [37], элементов сложных объектов.

Trie-дерево является инструментом поиска во множестве последовательностей. Хранение множества последовательностей, при необходимости, организуют во внешней по отношению к древовидной структуре дополнительно выделенной области памяти либо в листовых

вершинах. Тогда как возможность хранения последовательностей внутри древовидной структуры позволила бы создавать словари, обеспечивающие: скоростной поиск; повышение скорости добавления и удаления элементов; сокращение затрат памяти на хранение.

Рассмотрим trie-дерево [2, 3]. В этой структуре максимальное число потомков вершины ограничено размером алфавита, которому принадлежат строки, а отдельной строке соответствует ветвь от корня к листовой вершине. Trie-дерево имеет:

– внутренние вершины, у которых как минимум два потомка;
– листья, в которых хранятся последовательности [44].

Trie-дерево является сильноветвящимся деревом.

5. Проблемы сильноветвящихся деревьев. Одной из проблем сильноветвящихся деревьев является проблема размера вершины. Допустим, дерево цифрового поиска хранит последовательности, элементы которых принадлежат алфавиту A из τ элементов. Если в каждой вершине для ссылки на дочерние элементы хранить массив из τ элементов (по одному для каждого элемента алфавита), то «лишние» затраты памяти будут чрезмерно большими, так как большая часть будет «пустыми указателями». Если же в массиве ссылок на дочерние элементы хранить только «непустые» ссылки, то вершины дерева будут отличаться друг от друга по размеру. А это затрудняет хранение вершин в структурах данных с произвольным доступом (например, при их хранении в файле).

Ещё одной проблемой описания древовидных структур в классических и современных учебниках является использование для ссылки на вершины аппарата указателей (ссылочных структур). Например, узел дерева бинарного поиска [2] описан структурой NodeDesc, приведенной в листинге 1.

```
TYPE Node = POINTER TO NodeDesc;
TYPE NodeDesc = RECORD op: CHAR; left, right: Node END;
```

Листинг 1. Структура, описывающая дерево бинарного поиска

С деревьями на основе аппарата указателей проблематично выполнять операции без их загрузки в оперативную память компьютера. В то же время, в теории графов общепринятая практика – перенумеровать вершины и ссылаться на них по номеру. В графовых базах данных для ссылки на вершину также используют идентификаторы вершин.

6. Проблемы сопоставления рядов алгоритмом Dynamic Time Warping на примере задачи распознавания речевых команд.

В рамках этой задачи фактически решают задачу классификации ряда на основе меры расхождения распознаваемого и эталонного рядов. Алгоритм сопоставляет распознаваемую команду со всеми эталонами речевых команд. При этом распознаваемая речевая команда R и каждый эталон E являются последовательностями векторов признаков (рядами):

$$E = \mathbf{e}_1, ..., \mathbf{e}_j, ..., \mathbf{e}_n, \; R = \mathbf{r}_1, ..., \mathbf{r}_i, ..., \mathbf{r}_m, \tag{1}$$

полученными в результате предварительной обработки оцифрованного звука, который соответствует произнесению речевой команды.

Меру расхождения $d(E,R)$ находят следующим образом [24]. Вычисляют матрицу расстояний $\mathbf{D}$ между элементами $\mathbf{e}_i$ и $\mathbf{r}_j$:

$$\mathbf{D} = \left\| d_{ij} \right\|_{n \times m}, \; d_{ij} = d(e_i, r_j). \tag{2}$$

Далее, по матрице $\mathbf{D}$ вычисляют элементы DTW-матрицы выравнивания $\mathbf{K} = \left\| k_{ij} \right\|_{n \times m}$ по рекуррентным формулам:

$$\begin{aligned} & k_{11} = d_{11}; \\ & k_{1j} = d_{1j} + k_{1(j-1)}, j = \overline{2,n}; \\ & k_{i1} = d_{i1} + k_{(i-1)1}, i = \overline{2,m}; \\ & k_{ij} = d_{ij} + \min(k_{(i-1)(j-1)}, k_{i(j-1)}, k_{(i-1)j}), i = \overline{2,m}, \; j = \overline{2,n}. \end{aligned} \tag{3}$$

Мерой расхождения между E и R является значение k_{nm}:

$$d(E,R) = k_{nm}. \tag{4}$$

Полученная DTW-матрица $\mathbf{K}$, позволяет определить путь выравнивания – множество пар индексов соответствующих друг другу элементов $\mathbf{e}_{p_h}$ и $\mathbf{r}_{q_h}$ рядов E и R [24]:

$$M = \{(p_h, q_h)\}_{h=1}^{H}, \tag{5}$$

где H – количество пар в пути выравнивания. Эти пары определяют итеративно по формулам [24]:

$$\begin{aligned} &p_H = n, q_H = m \\ &(p_{h-1}, q_{h-1}) = \arg\min_{(p',q')\in P_h}(k_{p'q'}) \\ &P_h = \{(p',q') : p_h - 1 \le p' \le p_h, q_h - 1 \le q' \le q_h, (p' \ne p_h \vee q' \ne q_h)\} \\ &p_1 = 1, q_1 = 1 \end{aligned} . \quad (6)$$

Мера расхождения $d(E,R)$ представима также в виде суммы:

$$d(E,R) = \sum_{h=1}^{H} d(e_{p_h}, r_{q_h}) . \quad (7)$$

Словарь классов речевых команд W – это множество эталонов:

$$W = \left\{E_y\right\}_{y=1}^{z}, \quad (8)$$

а целевая функция находит номер класса y, между эталоном которого E_y и распознаваемой речевой командой R мера расхождения $d(E_y,R)$ минимальна:

$$res = \arg\min_{y=\overline{1,z}} d(E_y,R) . \quad (9)$$

Проблемы, возникающие при сопоставлении рядов с использованием алгоритма DTW [31]:

1) проблема пропуска алгоритмом DTW отличающихся элементов и учёта сходных (её пытаются решать введением ограничений на путь выравнивания, но это не всегда приносит ожидаемый эффект);

2) проблема длин эталонов при использовании DTW: если длина одного из эталонов значительно меньше длин остальных, мера расхождения от него до распознаваемого ряда может оказаться минимальной;

3) проблема вычислительной сложности DTW: вычислительная сложность классификации на основе алгоритма DTW

ряда длины m пропорциональна величине: $m \cdot \sum n_y$, где n_y – длина эталона с номером y в словаре классов речевых команд (8).

Решение проблемы длин эталонов за счёт нормировки меры расхождения по длине диагонали DTW-матрицы предложено в работе [31]. В данной работе предложены решения проблем пропуска алгоритмом DTW отличающихся элементов и учёта сходных и вычислительной сложности DTW.

7. Формальное описание усовершенствованного trie-дерева с учетом решения проблем сильноветвящихся деревьев. Отличия [39] усовершенствованного trie-дерева [36] от trie-дерева [2, 3]:

1) усовершенствованное trie-дерево содержит массив ссылок на вершины окончания строк;

2) вершины усовершенствованного trie-дерева имеют одинаковую структуру, в отличие от разделения на внутренние и листовые вершины у trie-дерева;

3) вершины усовершенствованного trie-дерева содержат ссылки на родительскую вершину и идентификатор строки, оканчивающейся в вершине;

4) в вершинах усовершенствованного trie-дерева хранится ссылка на множество вершин-потомков, что обеспечивает фиксированный размер вершины и позволяет хранить вершины в структурах данных произвольного доступа, а также ссылаться на вершины и множества вершин с помощью их номеров вместо использования аппарата указателей.

Пусть G – множество последовательностей, элементы которых принадлежат алфавиту A символов (или идентификаторов), $|A| = \tau$. Представление G в виде усовершенствованного trie-дерева:

$$\begin{aligned} &(O, Ch, X) \\ &O = \{\mathbf{o}_j\}, \mathbf{o}_j = (o_1^j, o_2^j, o_3^j, o_4^j) \\ &Ch = \{\mathbf{ch}_k\} \\ &X = \{x_i\}, x_i = j : o_2^j = i, i = \overline{1, z}, z = |G| \end{aligned}, \quad (10)$$

где O – массив векторов, описывающих вершины усовершенствованного trie-дерева; $o_1^j \in A$ – элемент последовательности; o_2^j – идентификатор последовательности, завершающейся в вершине номер j; o_3^j – номер в упорядоченном

множестве Ch массива, хранящего номера вершин-потомков вершины $\mathbf{o}_j$; o_4^j – номер родительской вершины для вершины $\mathbf{o}_j$; Ch – упорядоченное множество массивов $\mathbf{ch}_k$; $\mathbf{ch}_k$ – массив номеров вершин-потомков некоторой вершины ($ch_{o_3^j}$ – массив номеров вершин-потомков вершины $\mathbf{o}_j$); X – массив номеров вершин, в которых оканчиваются последовательности; x_i – номер j вершины $\mathbf{o}_j$, у которой поле o_2^j (идентификатор последовательности, завершающейся в вершине номер j) равно i .

Усовершенствованное trie-дерево является представлением словаря строк для хранения и поиска [36], так как для него определены функции получения: 1) идентификатора строки по её написанию; 2) строки по её идентификатору. Структура данных в программной реализации усовершенствованного trie-дерева описана в работе [39].

Фиксированный размер вершины позволяет хранить и обрабатывать вершины дерева в структурах данных с произвольным доступом (например, файлах); обрабатывать деревья, частично загруженные в оперативную память.

В проведенном эксперименте фиксированный размер вершины и использование идентификаторов вершин вместо аппарата указателей на словаре словоформ русского языка, содержащем около 2 млн. уникальных строк, увеличило на 40% скорость загрузки древовидной структуры из файла. Данный результат получен следующим образом. Измерено время загрузки усовершенствованного trie-дерева. Затем усовершенствованное trie-дерево модифицировано – изменён способ ссылки на вершины: вместо использования массивов и идентификаторов вершин применён аппарат указателей. После этого измерено время загрузки модифицированного дерева.

Исследования [38, 39], выполненные на множестве словоформ русского языка (около 2 млн. строк), показали, что предложенная иерархическая структура по сравнению с использованием массива с индексной таблицей уменьшает затраты памяти на 20%, времени на выполнение операций: поиска – на 37%, добавления и удаления – в 24 и 380 раз, соответственно.

Требует пояснения столь существенный прирост скорости операций добавления и удаления. Он связан с тем, что строка, попавшая в trie-дерево, никогда не меняет свой идентификатор. Точно так же, как вершина и массив вершин-потомков. Среди идентификаторов строк, вершин и массивов вершин-потомков есть

предопределённое значение 0, которое указывает на отсутствие ссылки. Значение поля $o_1^j = 0$ указывает на отсутствие символа в вершине $\mathbf{o}_j$. Например, в представлении (10) у корневой вершины $\mathbf{o}_1$ номер родительской вершины $o_4^1 = 0$ и номер символа вершины $o_1^1 = 0$; если в вершине $\mathbf{o}_j$ не оканчивается ни одна строка, то идентификатор строки, завершающейся в вершине номер j, – поле $o_2^j = 0$; если вершина $\mathbf{o}_j$ не имеет потомков, то идентификатор массива вершин-потомков $o_3^j = 0$.

Когда нужно удалить строку с номером i, удаляют фрагмент ветви, которая оканчивается в вершине с номером x_i, затем присваивают $x_i = 0$ и добавляют i в массив идентификаторов удаленных строк *DeletedStrings* [39]. Позже, когда понадобится добавить строку, идентификатор новой строки будет выбран из массива *DeletedStrings*, если он не пуст. Само нулевое значение остаётся в массиве X.

Аналогично, когда нужно удалить массив потомков массивов $\mathbf{ch}_k$, этот массив очищают, ссылку на него в вершине обнуляют и заносят номер k в массив номеров удалённых массивов вершин-потомков *DeletedChArr* [39]. Сам пустой массив $\mathbf{ch}_k$ остаётся во множестве Ch.

Когда нужно удалить вершину с номером j, то вектору $\mathbf{o}_j$ присваивают значение $\mathbf{o}_j = (0,0,0,0)$, предварительно удалив ссылку на неё из массива номеров вершин-потомков, самих вершин-потомков и массива номеров вершин, в которых оканчиваются последовательности. Заносят номер j в массив номеров удалённых вершин *DeletedKnots* [39]. Сам обнулённый вектор $\mathbf{o}_j$ остаётся во множестве O.

То есть, при операциях добавления и удаления строк в усовершенствованном trie-дереве минимизируются операции выделения памяти и перемещения в памяти больших фрагментов данных, как происходит при удалении элемента в начале массива.

8. Пофонемное распознавание речевых команд как задача классификации ряда на множестве последовательностей элементов сложных объектов с применением DTW. Фонетика рассматривает речь, в т. ч. слова и последовательности слов, как

последовательность фонетических единиц. В качестве фонетических единиц в данной работе используем разновидности звучания фонем, которые называют аллофонами и обозначают транскрипционными символами:

$$A = \{a_i\}_{i=1}^{\tau}. \quad (11)$$

Каждому аллофону соответствует класс звуков. Следует отметить, что эти классы звуков достаточно размыты в пространстве акустических признаков и имеют значительные пересечения.

Фонетическая транскрипция – это последовательность $T = t_1,..,t_\kappa,...,t_l$ транскрипционных символов $t_\kappa \in A$, где l – длина транскрипции T. Она обозначает последовательность аллофонов, из которых состоит слово или последовательность слов. Фонетическую транскрипцию речевой команды (один и более вариантов) можно построить по её написанию. То есть, фонетическая транскрипция речевой команды обозначает последовательность классов звуков, которой соответствует произнесение этой команды. При распознавании оцифрованный звук с произнесением речевой команды преобразуют в последовательность векторов признаков (ряд). По этому ряду необходимо найти последовательность классов звуков, составляющих соответствующую речевую команду.

Определение границ элементов в распознаваемом ряду по их границам в эталоне. Проблемой пофонемного распознавания является то, что границы аллофонов речевой команды внутри распознаваемого ряда $R = \mathbf{r}_1,...,\mathbf{r}_i,...,\mathbf{r}_m$ неизвестны. Для определения этих границ (сегментации распознаваемого ряда) предложено выполнять «условную сегментацию» [30] следующим образом. Распознаваемый ряд R сегментируют, исходя из предположения, что он соответствует последовательности классов звуков, которая обозначена транскрипцией T. То есть, каждой транскрипции будет соответствовать свой набор границ сегментов. Границы классов звуков (аллофонов) в R находят последовательно, исходя из предположений: 1) начало R совпадает с началом первого аллофона речевой команды; 2) при известной левой границе κ-го аллофона в R, его правую границу определяют, используя эталоны κ-го и $(\kappa+1)$-го аллофонов и считая, что элемент $\mathbf{r}_j$ ряда R относится к κ-му аллофону до тех пор, пока он не станет «ближе» к эталону $(\kappa+1)$-го аллофона, чем к эталону κ-го.

В работе [31] с помощью пути выравнивания M (4) пары рядов E и R (1), соответствующих одной и той же последовательности аллофонов (в работе [24] – разным произнесениям одного слова), по известному множеству границ начала аллофонов $B = \{b_1,...,b_\kappa,...,b_l\}$ в E получают соответствующие границы $B' = \{b'_1,...,b'_\kappa,...,b'_l\}$ в R.

Традиционно, путь выравнивания M строят при обратном проходе DTW-матрицы (6). Работа [31] предлагает алгоритмы, позволяющие вычислить путь выравнивания и меру расхождения при прямом проходе DTW-матрицы.

Мера расхождения между парой последовательностей векторов признаков, соответствующих одной последовательности классов звуков. Назовём фрагментом $E[x;y]$ ряда E последовательность $\mathbf{e}_x, \mathbf{e}_{x+1}, \ldots, \mathbf{e}_y$. Аналогично определим фрагмент $R[x';y']$. Пусть E и R соответствуют произнесениям слова с транскрипцией $T = t_1,..,t_\kappa,...,t_l$, а фрагменты $E[b_\kappa; b_{\kappa+1}-1]$ и $R[b'_\kappa; b'_{\kappa+1}-1]$ представляют в E и R аллофон t_κ. Это позволяет найти меру расхождения между E и R путём сопоставления их фрагментов, соответствующих отдельным аллофонам. Действительно, величина (4) представима как сумма мер расхождения между фрагментами E и R, соответствующими одному аллофону [31]:

$$d(E,R) = \sum_{k=1}^{l-1} d(E[b_\kappa; b_{\kappa+1}-1], R[b'_\kappa; b'_{\kappa+1}-1]) + d(E[b_l; n], R[b'_l; m]). \quad (12)$$

Мера расхождения с эталонной последовательностью, синтезированной по последовательности классов звуков. Напомним, что по правилу «условной сегментации» для аллофона, не являющегося последним аллофоном речевой команды, его правую границу необходимо определять, используя вектора признаков, относящиеся к κ-му и $(\kappa+1)$-му аллофонам. Т.е., нужно создавать эталоны для пар соседних аллофонов.

Пусть κ-й и $(\kappa+1)$-й аллофоны слова принимают значения g и h из алфавита A, а эталон аллофона g, за которым следует аллофон h, Et_g^h является последовательностью векторов (рядом) из c_{gh} элементов:

$$Et_g^h = \mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_{c_{gh}-c'}, \ldots, \mathbf{e}_{c_{gh}}. \quad (13)$$

Причём $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_{c_{gh}-c'}$ описывает аллофон g и межфонемный переход между аллофонами g и h (эта часть учитывается при подсчёте меры расхождения), а $\mathbf{e}_{c-c'+1}, \ldots, \mathbf{e}_c$ относится к аллофону h. Понадобятся также эталоны последних аллофонов речевых команд. Обозначим эталон аллофона g, являющегося последним в речевой команде, Et_g^{λ}. Он также представлен рядом (13).

Обозначим через $\Diamond$ операцию конкатенации (соединения) фрагментов последовательностей и рядов. Введённые выше обозначения, позволяют рассматривать эталонный ряд E как эталон, синтезированный путём конкатенации фрагментов эталонов не последних аллофонов $Et_{t_\kappa}^{t_{\kappa+1}}[1; c_{t_\kappa t_{\kappa+1}} - c']$, $\kappa = \overline{1, l-1}$ речевой команды $T = t_1, .., t_\kappa, ..., t_l$ с добавлением эталона последнего аллофона Et_g^{λ}:

$$E = Et_{t_1}^{t_2}[1; c_{t_1 t_2} - c'] \Diamond Et_{t_2}^{t_3}[1; c_{t_1 t_2} - c'] \Diamond \ldots \Diamond Et_{t_{l-1}}^{t_l}[1; c_{t_{(l-1)} t_l} - c'] \Diamond E_{t_l}^{\lambda}. \quad (14)$$

Приведём алгоритм вычисления меры расхождения $d_1(Et_{t_\kappa}^{t_{\kappa+1}}, R[b'_\kappa; b'_{\kappa+1}], u_\kappa)$ от эталона $Et_{t_\kappa}^{t_{\kappa+1}}$ (13) до соответствующего фрагмента $R[b'_\kappa; b'_{\kappa+1}]$, с сохранением границы $b'_{\kappa+1}$ начала $(\kappa+1)$-го аллофона и длины u_κ эталона $Et_{t_\kappa}^{t_{\kappa+1}}$ (Алгоритм 1), который является производным от алгоритма, работы [31]. В рамках данного алгоритма длину $c_{t_\kappa t_{\kappa+1}}$ эталона $Et_{t_\kappa}^{t_{\kappa+1}}$ обозначим через c.

Считаем b'_κ – левую границу κ-го аллофона в R, известной. Левую границу $b'_{\kappa+1}$ следующего за ним $(\kappa+1)$-го аллофона в R получим в процессе вычисления искомой меры расхождения. Для вычисления понадобится DTW-матрица $\mathbf{K}$ размерности $c \times m$.

Заполняем её от элемента с индексом $(1,1)$, который является первым «рабочим элементом». Заполнение происходит, пока рабочим элементом не станет элемент $k_{c\,q}$ строки с номером c. При анализе рабочего элемента $k_{p_h\,q_h}$ используем элементы столбцов q_h и (q_h+1).

1. *Инициализация* // вычислить 2 столбца DTW-матрицы $\mathbf{K}$

1.1 $k_{11} := d(\mathbf{e}_1, \mathbf{r}_{b'_\kappa})$, $k_{s1} := d(\mathbf{e}_s, \mathbf{r}_{b'_\kappa}) + k_{(s-1)1}$, $s = \overline{2,c}$

1.2 $k_{12} := d(\mathbf{e}_1, \mathbf{r}_{b'_\kappa+1}) + k_{11}$

1.3 $k_{s2} := d(\mathbf{e}_s, \mathbf{r}_{b'_\kappa+1}) + \min(k_{(s-1)2}, k_{(s-1)1}, k_{s1}), s = \overline{2,c}$

1.4 $f := 1, j := b'_\kappa$, $p := 1, q := b'_\kappa$

2. *Пока* $(p \le c) \wedge (q \le m)$ *выполнять*

2.1 *если* $q = j + 1$ *то* //вычислить $(q+1)$-й столбец матрицы $\mathbf{K}$

2.1.1 $k_{1(q+1)} := d(\mathbf{e}_1, \mathbf{r}_{q+1}) + k_{1q}$

2.1.2 $k_{s(q+1)} := d(\mathbf{e}_s, \mathbf{r}_{q+1}) + \min(k_{(s-1)q}, k_{(s-1)(q+1)}, k_{s(q+1)}), s = \overline{2,c}$

2.2 $f := p, j := q$

2.3 $(p,q) = \arg\min_{(p',q') \in P_{fj}} (k_{p'q'})$, где $P_{fj} = \{(p',q')\}$,

$f \le p' \le f+1, j \le q' \le j+1, p' \le n, q' \le m, (p' = f \vee q' = j)$

2.4 *если* $(p = c) \wedge (q = j)$ *то* $p := f+1$, $q := j+1$

3. *Если* $(p < c) \wedge (q = m)$ // достигли конца последовательности R

3.1 *то* ***Результат:***

$b'_{\kappa+1} := m$ – номер вектора начала $(\kappa+1)$-го аллофона

$u_\kappa := (c - c')$ – длина эталона аллофона t_κ

$d_1(Et^{t_{\kappa+1}}_{t_\kappa}, R[b'_\kappa; b'_{\kappa+1}], u_\kappa) := -1$ («отказ от распознавания»)

3.2 *иначе* // достигли конца эталона $Et^{t_{\kappa+1}}_{t_\kappa}$

3.2.1 $f := p, j := q$

3.2.2 *Пока* $(p > 1) \wedge (q > b_\kappa) \wedge (p \ge c - c')$ *выполнять*

$(p,q) = \arg\min_{(p',q') \in P_{fj}} (k_{p'q'})$, $P_{fj} = \{(p',q')\}$, где

$f - 1 \le p' \le f, j - 1 \le q' \le j, p' \ge 1, q' \ge b_\kappa, (p' = f - 1) \vee (q' = j - 1)$

3.2.3 ***Результат:***

$b'_{\kappa+1} := q+1$ – номер вектора начала $(\kappa+1)$-го аллофона

$u_\kappa := (c - c')$ – длина эталона аллофона t_κ

$d_1(Et^{t_{\kappa+1}}_{t_\kappa}, R[b'_\kappa; b'_{\kappa+1}], u_\kappa) := k_{pq}$ – искомая мера расхождения

Алгоритм 1. Мера расхождения $d_1(Et^{t_{\kappa+1}}_{t_\kappa}, R[b'_\kappa; b'_{\kappa+1}], u_\kappa)$

Поэтому алгоритм начнём с вычисления первых двух столбцов, далее при каждом изменении номера столбца рабочего элемента будем вычислять значение следующего столбца.

Если на некотором шаге итерации индексы рабочего элемента приняли значение (p,m) и $p < c$, то делаем вывод о несоответствии ряда R и последовательности аллофонов T, поскольку достигли границы R, но не достигли границы не последнего аллофона. В этом случае присвоим искомой мере расхождения значение –1 и будем говорить, что произошёл «отказ от распознавания».

Когда индексы рабочего элемента достигают значение (c,q), то рассмотрен последний вектор эталона пары аллофонов. Вектор эталона $Et_{t_\kappa}^{t_{\kappa+1}}$ с индексом c уже принадлежит аллофону $t_{\kappa+1}$. Значит, вектор $\mathbf{r}_q$ ряда R принадлежит аллофону $t_{\kappa+1}$, т.е. $b'_{\kappa+1} \le q$.

Найдём границу $b'_{\kappa+1}$, для чего выполним обратный проход по DTW-матрице, пока текущим элементом обратного прохода не станет элемент $k_{(c-c')q_1}$. Это и есть искомая мера расхождения, а значение (q_1+1) является – $b'_{\kappa+1}$ левой границей $(\kappa+1)$-го аллофона в R.

В экспериментальной части данной работы [31] для константы c' использовано значение 2.

Алгоритм 1 используют для вычисления меры расхождения от не последнего аллофона речевой команды до фрагмента R.

Опишем меру расхождения от эталона последнего аллофона до фрагмента R. Для этого обозначим паузу, следующую за последним аллофоном слова, специальным символом λ и введём его в алфавит аллофонов A. Обозначим через $d_1'(Et_{t_l}^{\lambda}, R[b_l;m], u_l)$ меру расхождения между эталоном последнего аллофона Et_{tl}^{λ} слова и фрагментом $R[b_l;m]$. Вычислим $d_1'(Et_{t_l}^{\lambda}, R[b_l;m], u_l)$ как значение элемента $k_{(c-c')m}$ DTW-матрицы $\mathbf{K} = \left\| k_{ij} \right\|_{(c-c')\times m}$ по рекуррентным формулам (3); u_l присвоим значение $(c-c')$.

Мера расхождения между распознаваемой последовательностью и последовательностью классов звуков. Выполним постановку задачи пофонемного распознавания как задачи классификации ряда на множестве последовательностей аллофонов и обоснуем возможность её решения.

Как отмечено выше, для определения границы между соседними аллофонами нужны эталоны для пар соседних аллофонов $Et_{t_к}^{t_{к+1}}$ (13), где $t_к$ текущий аллофон, а $t_{к+1}$ – следующий аллофон.

Пусть образ пары аллофонов $\mathrm{E}_{a_i}^{a_j}$ представлен множеством эталонов $Et_{a_i}^{a_j}$, являющихся рядами (13):

$$\mathrm{E}_{a_i}^{a_j} = \left\{ Et_{a_i}^{a_j} \right\},\ Et_{a_i}^{a_j} = \mathbf{e}_1, \ldots, \mathbf{e}_{c-c'}, \ldots \mathbf{e}_c . \tag{15}$$

Отметим, что длина c у разных эталонов может отличаться, значение c' является константой.

Назовём алфавитом образов аллофонов пару упорядоченных множеств $\langle A, \mathrm{E} \rangle$, содержащую множество обозначений аллофонов A с помощью транскрипционных символов и множество образов пар аллофонов (текущий-следующий аллофон) E:

$$\langle A, \mathrm{E} \rangle,\ \ A = \{a_i\}, \mathrm{E} = \{\mathrm{E}_{a_i}^{a_j}\} . \tag{16}$$

Сформируем словарь классов речевых команд W как совокупность алфавита образов аллофонов $\langle A, \mathrm{E} \rangle$ и словаря строк $G = \{T_y\}_{y=1}^{z}$ – множества фонетических транскрипций речевых команд:

$$W = \langle A, \mathrm{E}, G \rangle . \tag{17}$$

При классификации ряда R (1) нужно найти номер y транскрипции T_y, определяющей последовательность образов пар аллофонов, для которой мера расхождения с R минимальна:

$$res = \arg\min_{y=\overline{1,z}} F(T_y, R) : F(T_y, R) \neq -1 , \tag{18}$$

где $F(T_y, R)$ – функционал для вычисления меры расхождения между последовательностью образов пар аллофонов, полученных по фонетической транскрипции T_y, и распознаваемым рядом R.

Функционал $F(T_y, R)$ позволяет выполнить отказ от распознавания – в случае, когда не удалось найти фрагменты для всех образов пар аллофонов из транскрипции T_y в распознаваемом ряду R, он возвращает значение «–1».

Для разработки функционала $F(T, R)$ потребуется ввести функционалы для вычисления меры расхождения между образом пары аллофонов и соответствующим фрагментом ряда R.

Меру расхождения $\Phi(\mathrm{E}_{t_\kappa}^{t_{\kappa+1}}, \mathrm{R}[b'_\kappa; b'_{\kappa+1}], u_\kappa)$ от образа пары аллофонов $\mathrm{E}_{t_\kappa}^{t_{\kappa+1}}$ до соответствующего фрагмента $R[b'_\kappa; b'_{\kappa+1}]$, с сохранением границы $b'_{\kappa+1}$ начала $(\kappa+1)$-го аллофона и длины u_κ того эталона nEt, для которого минимально отношение меры расхождения к длине диагонали DTW-матрицы и не произошло отказа от распознавания, вычисляем следующим образом.

Из эталонов $Et \in \mathrm{E}_{t_\kappa}^{t_{\kappa+1}}$ сформируем множество $\widehat{\mathrm{E}_{t_\kappa}^{t_{\kappa+1}}}$ эталонов, для которых при вычислении $d_1(Et, R[b'_\kappa; b'_{\kappa+1}], u_\kappa)$ не происходит отказ от распознавания:

$$\widehat{\mathrm{E}_{t_\kappa}^{t_{\kappa+1}}} = \{Et\} : \left(Et \in \mathrm{E}_{t_\kappa}^{t_{\kappa+1}}\right) \wedge \left(d_1(Et, R[b'_\kappa; b'_{\kappa+1}], u_\kappa) \neq -1\right). \tag{19}$$

Из этого множества выберем эталон nEt, с минимальным отношением меры расхождения к длине диагонали DTW-матрицы:

$$nEt = \underset{Et \in \widehat{\mathrm{E}_{t_\kappa}^{t_{\kappa+1}}}}{\arg\min} \left(\frac{d_1(Et, R[b'_\kappa; b'_{\kappa+1}], u_\kappa)}{\sqrt{(u_\kappa)^2 + (b'_{\kappa+1} - b'_\kappa)^2}} \right). \tag{20}$$

Если множество $\widehat{\mathrm{E}_{t_\kappa}^{t_{\kappa+1}}} = \varnothing$, происходит отказ от распознавания.

$$\Phi(\mathrm{E}_{t_\kappa}^{t_{\kappa+1}}, R[b'_\kappa; b'_{\kappa+1}], u_\kappa) = \begin{cases} d_1(nEt, R[b'_\kappa; b'_{\kappa+1}], u_\kappa), \widehat{\mathrm{E}_{t_\kappa}^{t_{\kappa+1}}} \neq \varnothing \\ -1 \qquad\qquad\qquad\qquad , \textit{иначе} \end{cases}, \tag{21}$$

где $\widehat{\mathrm{E}_{t_\kappa}^{t_{\kappa+1}}}$ – множество эталонов пар аллофонов (19), для которых при вычислении $d_1(Et, R[b'_\kappa; b'_{\kappa+1}], u_\kappa)$ не происходит отказ от распознавания; nEt – эталон из множества $\widehat{\mathrm{E}_{t_\kappa}^{t_{\kappa+1}}}$, для которого минимально отношение меры расхождения к длине диагонали DTW-матрицы (20).

При вычислении меры расхождения $\Phi'(\mathrm{E}_{t_l}^{\lambda}, \mathrm{R}[\mathrm{b}_1; \mathrm{m}], \mathrm{u}_\kappa)$ от образа пары последнего аллофона и завершающей паузы $\mathrm{E}_{tl}^{\lambda}$ до соответствующего фрагмента $R[b'_l; m]$ отказа от распознавания не происходит. Вычисление меры расхождения с сохранением длины u_l того эталона, для которого минимальна мера расхождения делённая на длину диагонали DTW-матрицы, происходит следующим образом:

$$\Phi'(\mathrm{E}_{tl}^{\lambda}, R[b'_l; m], u_l) = d'_1(Et_{tl}^{\lambda}, R[b'_l; m], u_l),$$
$$Et_{tl}^{\lambda} = \underset{Et_{tl}^{\lambda} \in \mathrm{E}_{tl}^{\lambda}}{\arg\min} \left(\frac{d'_1(Et_{tl}^{\lambda}, R[b'_l; m], u_l)}{\sqrt{(u_l)^2 + (m - b'_\kappa)^2}} \right). \quad (22)$$

Функционал $F(T, R)$ для вычисления меры расхождения между последовательностью образов пар аллофонов, полученных по фонетической транскрипции T и рядом R, позволяющий выполнить отказ от распознавания:

$$F(T,R) = \begin{cases} -1, & \exists\, \kappa = \overline{1, l-1} : \Phi(\mathrm{E}_{t_\kappa}^{t_{\kappa+1}}, R[b'_\kappa; b'_{\kappa+1}], u_\kappa) = -1 \\ \dfrac{\sum_{\kappa=1}^{l-1} \Phi(\mathrm{E}_{t_\kappa}^{t_{\kappa+1}}, R[b'_\kappa; b'_{\kappa+1}], u_\kappa) + \Phi'(\mathrm{E}_{tl}^{\lambda}, R[b'_l; m], u_l)}{H}, & \textit{иначе} \end{cases}, \quad (23)$$

$$b_1 = 1, H = \sqrt{U^2 + m^2}, U = \sum_{\kappa=1}^{l} u_\kappa.$$

Благодаря возможности выполнить отказ от распознавания в случае невозможности найти границы всех образов $\mathrm{E}_{t_\kappa}^{t_{\kappa+1}}$ пар аллофонов из транскрипции $T = t_1, .., t_\kappa, ..., t_l$ в R, предложенная мера

расхождения $F(T,R)$ учитывает фонетический состав речевой команды (транскрипцию T) и тем самым решает проблему пропуска алгоритмом DTW отличающихся элементов и учёта сходных.

9. Разработка метода классификации ряда на множестве последовательностей элементов сложных объектов с применением усовершенствованного trie-дерева на примере пофонемного распознавания с иерархическим представлением словаря классов речевых команд. Приведенный выше метод пофонемного распознавания применим только для распознавания речевых команд из небольшого словаря, так как вычисление функционала $F(T,R)$ выполняется отдельно для каждой транскрипции речевой команды.

Преобразуем словарь классов речевых команд W (17), заменив в нём представление множества фонетических транскрипций речевых команд $G=\{T_y\}$ на усовершенствованное trie-дерево (10):

$$W=\langle A,\mathrm{E},O,Ch,X\rangle. \tag{24}$$

Разработаем метод пофонемного распознавания, используя это иерархическое представление словаря классов речевых команд. Поставим в соответствие вершине $\mathbf{o}_j$ набор значений:

1) b_j – левая граница в R аллофона, которому соответствует вершина $\mathbf{o}_j$;

2) $dist_j$ – число, обозначающее найденную интегральную меру расхождения между фрагментом распознаваемой последовательностей $R[1;b_j]$ и начальной частью множества речевых команд, транскрипция которых хранится в ветви от корня дерева структуры до вершины $\mathbf{o}_j$;

3) len_j – суммарная длина эталонов цепочки аллофонов (в ветви от корня дерева до вершины $\mathbf{o}_j$), использованных при распознавании фрагмента ряда R, соответствующего этой цепочке.

$TW(k_i)$ – функция получения номера следующей вершины при обходе вершин дерева в глубину; $TW1(k_i)$ – функция получения номера следующей вершины при обходе вершин дерева вверх по ветви

до первого правого потомка. Функции $TW(k_i)$ и $TW1(k_i)$ возвращают номера вершин или 0.

$V = \{v_j\}_{j=1}^{z}$ – массив интегральных мер расхождения между распознаваемым рядом R и транскрипцией словаря речевых команд с идентификатором j.

Обозначим Y множество номеров вершин, в которых оканчиваются транскрипции распознанных команд.

Используем также промежуточные переменные tmp_d, u, Id.

Пофонемное распознавание R (1) выполняем путём обхода вершин (O, Ch, X) усовершенствованного trie-дерева (Алгоритм 2).

Пусть $N = |O|$ – количество вершин, а $z = |X|$ – количество последовательностей в дереве. Переменной k обозначим номер вершины, обрабатываемой на текущем шаге обхода дерева; вершины нумеруем с 1 ($\mathbf{o}_1$ – корневая вершина); значение 0 обозначает отсутствие искомой вершины в дереве. При инициализации: назначаем текущей вершиной корень дерева ($k := 1$); границей начала первого аллофона для всех последовательностей в дереве – 1-й вектор ряда ($b_k := 1$); очищаем множество Y; обнуляем b_j, $dist_j$, len_j, связанные с вершинами $\mathbf{o}_j$; обнуляем v_j, связанные с идентификаторами j последовательностей, хранимых в дереве. Выполняем обход вершин дерева, $\mathbf{o}_k$ – текущая вершина.

Возможна ситуация, когда вычисленная для вершины $\mathbf{o}_k$ граница начала аллофона больше или равна длине ряда R ($b_k \geq m$), или для вершины $\mathbf{o}_k$ произошел отказ от распознавания ($dist_k < 0$).

В этом случае последовательности образов аллофонов, начало которых соответствует фрагменту ветви от корня до вершины $\mathbf{o}_k$, не могут быть результатом распознавания ряда R – переходим к следующей вершине при обходе вверх по ветви до первого правого потомка.

Если в вершине оканчивается последовательность ($o_2^k \neq 0$), то вычисляем меру расхождения $\Phi'(E_g^{\lambda}, R[b_k; m], u)$ от образа $\mathrm{E}_{\mathrm{g}}^{\lambda}$

последнего аллофона g, которому соответствует вершина $\mathbf{o}_k$ (п. 2.2 Алгоритма 2), до оставшегося фрагмента ряда $R[b_k; m]$.

Если нет отказа от распознавания (п. 2.3.2 Алгоритма 2), то внести результаты сопоставления с R последовательности образов аллофонов с идентификатором o_2^k в массив V и множество Y.

1. *Инициализация*
   1.1 $N := |O|; z := |X|;\ k := 1; b_k := 1; Y := \varnothing;$
   1.2 *Для* $j = \overline{1,N}$ *выполнять* $b_j := 0; dist_j := 0; len_j := 0;$
   1.3 *Для* $j = \overline{1,z}$ *выполнять* $v_j := -1;$
2. *Пока* $(k \neq 0)$ *выполнять*

н.ц.

   2.1 *Если* $(b_k \geq m) \vee (dist_k < 0)$ *то* *Перейти на шаг* 2.7;
   2.2 $g := o_1^k;$
   2.3 *Если* $o_2^k \neq 0$ *то*
   2.3.1 $tmp\_d := \Phi'(\mathrm{E}_g^{\lambda}, R[b_k; m], u);$
   2.3.2 *Если* $(tmp\_d > -1)$
   то $Id := o_2^k; Y := Y \cup Id; v_{Id} := \dfrac{dist_k + tmp\_d}{\sqrt{(len_k + u)^2 + m^2}};$
   2.4 *Для* $n \in ch_{o_3^k}$ *выполнять*

н.ц.

   2.4.1 $h := o_1^n;$
   2.4.2 $tmp\_d := \Phi(\mathrm{E}_g^h, R[b_k; b_n], u);$
   2.4.3 *Если* $(tmp\_d > -1)$ $tmp\_d := -1;$
   то $dist_n := dist_k + tmp\_d; len_n := len_k + u;$
   2.4.4 *иначе* $b_n := m; dist_n := -1;$

к.ц.

   2.5 $k := TW(k);$
   2.6 *Если* $(k = 0)$
   то перейти на шаг 2.7;
   иначе перейти на шаг 2.8;
   2.7 $k := TW1(k);$
   2.8 ;

к.ц.

3. ***Результат:*** $j := \arg\min_{i \in Y}(v_i) : v_i \neq -1$

Алгоритм 2. Пофонемное распознавание с иерархическим представлением словаря классов речевых команд и нормировкой по длине эталона

Для всех вершин $\mathbf{o}_n$, являющихся потомками вершины $\mathbf{o}_k$, вычислить меру расхождения $\Phi(\mathrm{E}_g^h, R[b_k; b_n], u)$ до образа E_g^h где g и h соответствуют вершинам $\mathbf{o}_k$ и $\mathbf{o}_n$. Фиксируем результаты в массивах B, $Dist$, Len с учетом того, имел ли место отказ от распознавания.

Переходим к следующей вершине дерева.

Предложенный метод позволяет анализировать общие начальные части последовательностей образов (классов объектов) единожды, за счёт чего обеспечивает ускорение классификации последовательности векторов признаков на множестве последовательностей образов.

Численные исследования разработанного метода проведены на трёх словарях речевых команд, которые обозначим «С1», «С2», «С3». Перечень речевых команд, использованных для создания эталонов пар аллофонов, обозначен «Ф». В качестве речевых команд использованы слова русского языка, причем выбран набор слов, являющийся сложным для классификации алгоритмом DTW (слова фонетически близки; некоторые слова являются фрагментом других; слова существенно отличаются по длине). Перечни слов указанных словарей и слов, использованных при обучении эталонов пар аллофонов, приведен в таблице 1. Символ «\» обозначает ударение.

Результаты численных исследований приведены в таблицах 2-3. Распознаванию подвергались все слова словаря. Обучающая и тестовая выборка не пересекались.

Все слова произнесены одним диктором на одном звукозаписывающем оборудовании и в схожих акустических условиях. Для распознавания на основе алгоритма DTW каждая речевая команда имела один эталон, построенный по одной аудиозаписи произнесения речевой команды. Для пофонемного распознавания использованы эталоны пар аллофонов, которые построены по вручную размеченным аудиозаписям слов из набора «Ф», который приведен в таблице 1.

В таблице 2 приведено количество верно распознанных слов для словарей С1, С2, С3 при использовании определённого метода классификации. На рисунке 1 показано отношение числа верно распознанных команд к числу предъявленных для распознавания.

Из таблицы 2 видно, что и у ранее предложенного метода пофонемного распознавания [31] и у разработанного метода пофонемного распознавания с применением усовершенствованного trie-дерева совпадает количество верно распознанных команд. Это объясняется тем, что оба метода вычисляют меру расхождения по одному и тому же функционалу.

Таблица 1. Содержимое словарей С1, С2, С3 и перечень слов «Ф»

Слово	С3	С2	С1	Ф	Слово	С3	С2	С1	Ф	Слово	С3	С2	С1	Ф
\ил				+	м\ена	+				прот\ест	+			
\унт	+	+	+	+	м\есто	+	+	+	+	прот\опать	+	+	+	+
алл\ель	+	+			м\етка	+	+			с\ель	+			
б\оль	+	+	+	+	м\етод	+	+			с\ено	+			
багат\ель	+	+			мет\елица	+	+	+	+	с\ет	+			
бал\ет	+				мет\ель	+	+			с\етка	+			
бал\етки	+			+	мет\ил	+				с\етовать	+			
балаб\олить	+	+			минов\ать	+	+	+	+	своров\ать	+	+	+	+
балов\ать	+				набалов\ать	+	+	+	+	сел\ен	+			
вал\ет	+	+			наворов\ать	+				синаг\ога	+			
воров\ать	+	+			надков\ать	+				сов\ать	+			
г\оголь	+	+	+	+	надыш\ать	+	+	+	+	сол\ило	+			
г\огот	+				нал\ив	+				сос\ед	+			
г\оль	+	+			налив\ать	+				сос\едка	+	+	+	+
г\опать	+				налинов\ать	+	+	+	+	ст\оль	+			
гал\ета	+				налом\ать	+		+	+	став\ать	+			
гал\етка	+				нас\едка	+	+	+	+	стогов\ать	+	+	+	+
гогот\ун	+	+			нас\ест	+	+	+	+	т\ент	+	+		
д\ив	+	+			насев\ать	+	+	+	+	т\ест	+	+		
дыш\ать	+	+	+		нат\опать	+	+	+	+	т\есто	+	+		
заболев\ать	+	+			од\ышка	+	+		+	т\иф	+			
забалов\ать				+	однол\етка	+	+	+	+	т\ога	+			
завод\ила	+	+	+	+	ол\ифа	+	+	+	+	т\оль	+	+		
задыш\ать	+	+	+	+	от\ель	+	+			т\опать	+	+		
залив\ать	+	+			отков\ать	+	+			т\опливо	+			
засев\ать	+	+			пал\етка	+	+			т\ополь	+	+		
зат\опать	+	+	+		паралл\ель	+	+	+	+	т\опот	+	+		
кад\ило	+				паров\ать	+	+			т\ун	+	+		
калев\ать	+	+			пат\ент	+	+			уг\одно				+
карав\ай	+				патов\ать	+	+			улом\ать				+
карот\ель	+	+	+	+	пл\ед	+	+			ф\и	+	+		
кив\ать	+	+	+	+	пл\ен	+	+	+	+	ф\ила	+	+		
килев\ать	+	+	+	+	пл\и	+	+			ф\ифа	+	+	+	+
кис\ель	+	+			плев\ать		+			фас\ет	+	+		
кис\ет	+	+	+	+	плинтов\ать	+	+			фас\етка	+	+		
ков\ать	+				побалов\ать	+	+			хал\иф	+	+	+	+
кол\ено	+	+	+	+	поворов\ать	+	+			хохот\ун	+	+	+	+
кол\ет	+	+	+	+	под\и	+	+			целов\ать	+	+	+	+
колт\ун	+	+	+	+	подков\ать	+								
л\ай	+	+	+	+	подыш\ать	+								
л\ен	+	+			пол\ено	+								
л\ента	+	+	+	+	пол\ив	+								
л\ето	+	+			полив\ать	+								
л\етом	+	+	+	+	полинов\ать	+								
л\и				+	полом\ать	+								
л\иф	+				пос\етовать	+								
линов\ать	+	+			пот\опать	+								
лод\ыжка	+	+	+	+	продыш\ать	+	+	+	+					
лом\ать	+	+	+	+	прол\ив	+								
м\ай	+	+	+	+	пролив\ать	+								
м\ать	+	+			пролинов\ать	+								
м\елево	+	+	+	+	пролом\ать	+								
м\ель	+	+			просев\ать	+								

Таблица 2. Количество верно распознанных речевых команд

Словарь	Обозначение	C1	C2	C3
	Количество команд в словаре	45	91	138
Метод	Пофонемное распознавание	44	82	121
	Пофонемное распознавание с применением усовершенствованного trie-дерева	44	82	121
	DTW со взвешиванием по длине диагонали матрицы выравнивания	32	53	76
	DTW без взвешивания	32	51	74

В проведенном эксперименте у методов пофонемного распознавания отношение числа верно распознанных команд к числу предъявленных для распознавания более, чем на ¼ превзошло тот же показатель метода DTW. Это объясняется тем, что выбран словарь, демонстрирующий недостатки DTW (более подробно этот вопрос рассмотрен в работе [32]). Словари с такими характеристиками можно использовать для анализа работы методов распознавания в худшем случае, но нельзя использовать в реальных задачах распознавания речевых команд, так как речевые команды должны быть хорошо различимы.

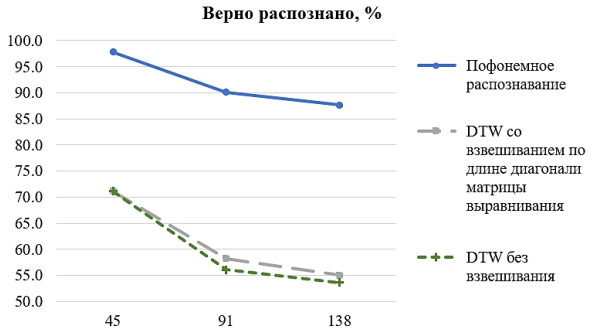

Рис. 1. Отношение числа верно распознанных команд к числу предъявленных для распознавания

Разработанный метод пофонемного распознавания с применением усовершенствованного trie-дерева отличается от предложенного ранее метод пофонемного распознавания [31] порядком, в котором вычисляют меры расхождения $\Phi(E_{t_к}^{t_{к+1}}, R[b'_к, b'_{к+1}], u_к)$ от образов пар аллофонов E множества

речевых команд до соответствующих фрагментов $R[b'_{\kappa}, b'_{\kappa+1}]$. Ранее предложенный метод сопоставлял с R каждую речевую команду отдельно. Разработанный метод общие начальные части речевых команд с соответствующими фрагментами R сопоставляет однократно, что снижает вычислительную сложность метода.

В таблице 3 показаны среднее время распознавания речевых команд в зависимости размера словаря и применённого метода.

Таблица 3. Среднее время распознавания речевых команд, мс

Словарь	Обозначение	C1	C2	C3
	Количество команд в словаре	45	91	138
Метод	Пофонемное распознавание	60	89	129
	Пофонемное распознавание с применением усовершенствованного trie-дерева	47	68	97
	DTW со взвешиванием по длине диагонали матрицы выравнивания	47	86	134
	DTW без взвешивания	47	86	137

На рисунке 2 отражено отношение времени распознавания различными методами ко времени распознавания разработанным методом.

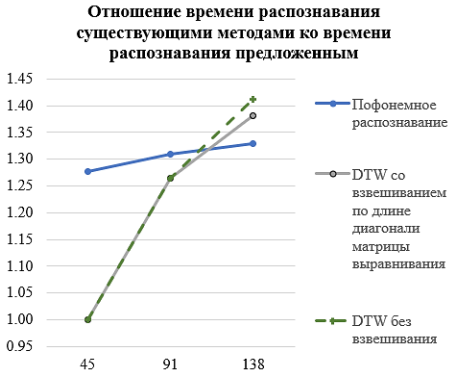

Рис. 2. Отношение времени распознавания различными методами ко времени распознавания разработанным методом

Для словаря C1 отношение времени распознавания методом DTW ко времени распознавания разработанным методом равно 1. Это можно объяснить тем, что количество речевого материала в эталонах метода DTW несколько меньше, чем в эталонах пар аллофонов,

используемых в пофонемном распознавании. Увеличение размера словаря влечёт увеличение количества речевого материала в эталонах метода DTW, что вызывает больший рост вычислительной сложности метода DTW, чем разработанного метода.

9. Заключение. Последовательности могут быть применены во многих областях. Прослеживается тенденция к обобщению задач из различных предметных областей как задач анализа последовательностей.

Впервые сформулированы две проблемы сильноветвящихся деревьев, которые решает усовершенствованное trie-дерево: проблема размера вершины, которая состоит в том, что вершины одного дерева имеют разный размер из-за разного числа вершин-потомков; использование для ссылки на вершины аппарата указателей. Практическая значимость решения этих проблем состоит в следующем. Фиксированный размер вершины и использование идентификаторов вершин вместо аппарата указателей позволяет: хранить и обрабатывать вершины дерева в структурах данных с произвольным доступом (например, файлах); обрабатывать деревья, частично загруженные в оперативную память. В проведенном эксперименте фиксированный размер вершины и использование идентификаторов вершин вместо аппарата указателей увеличило на 40% скорость полной загрузки из файла усовершенствованного trie-дерева. Выполнено формальное описание усовершенствованного trie-дерева с учетом решения проблем сильноветвящихся деревьев. Дано пояснение ранее полученным результатам о существенном приросте скорости операций добавления и удаления последовательностей в усовершенствованном trie-дереве относительно использования массива с индексной таблицей.

Разработан метод классификации ряда на множестве последовательностей элементов сложных объектов с применением усовершенствованного trie-дерева на примере пофонемного распознавания с иерархическим представлением словаря классов речевых команд. Продемонстрировано, что усовершенствованное trie-дерево является способом представления множеств последовательностей, который можно эффективно применять в задачах классификации на множестве последовательностей элементов сложных объектов при использовании аддитивной или мультипликативной мер подобия / различия.

Численные исследования подтвердили, что метод пофонемного распознавания речевых команд как задачи классификации ряда на множестве последовательностей элементов сложных объектов решает

проблему пропуска алгоритмом DTW отличающихся элементов и учёта сходных.

Показано, что разработанный метод классификации ряда на множестве последовательностей элементов сложных объектов с применением усовершенствованного trie-дерева решает проблему вычислительной сложности алгоритма DTW за счёт однократного сопоставления общих начальных частей последовательностей с соответствующими им фрагментами ряда.

Пофонемное распознавание с иерархическим представлением словаря классов речевых команд не претендует на то, чтобы конкурировать с современными средствами распознавания речи. Пофонемное распознавание выбрано как «модельная задача» из хорошо знакомой автору предметной области. С помощью предложенного метода в будущем предполагается решать задачи классификации на множестве последовательностей элементов сложных объектов в других предметных областях для случая, когда количество размеченных данных недостаточно для обучения скрытой марковской модели или нейронной сети.

Литература

1. Вирт Н. Алгоритмы и структуры данных. Новая версия для Оберона + CD // М.: ДМК Пресс. 2010. 272 с.
2. Кнут Д.Э. Искусство программирования. Т.3: Сортировка и поиск // М.: Вильямс. 2000. 832 с.
3. Briandais R. File searching using variable-length keys // Proc. Western Joint Computer Conf. 1959. pp. 295–298.
4. Гасфилд Д. Строки, деревья и последовательности в алгоритмах: Информатика и вычислительная биология // СПб.: Невский Диалект; БХВ-Петербург. 2003. 654 с.
5. Liao T.F., Bolano D., Brzinsky-Fay C., Cornwell B., Fasang A.E., Helske S., Piccarreta R., Raab M., Ritschard G., Struffolino E., Studer M. Sequence analysis: Its past, present, and future. Social science research. 2022. vol. 107. DOI: 10.1016/j.ssresearch.2022.102772.
6. Mathew S., Peat G., Parry E., Sokhal B.S., Yu D. Applying sequence analysis to uncover 'real-world' clinical pathways from routinely collected data: a systematic review. Journal of Clinical Epidemiology. 2024. vol. 166. DOI: 10.1016/j.jclinepi.2023.111226.
7. Громов В.А., Мазайшвили К.В., Заикин П.В., Николаев Е.Н., Бесчастнов Ю.Н., Зворыкина Е.И., Паринов А.А., Незнанов А.А. Различение хаотических и регулярных временных рядов для идентификации состояния артериовенозной фистулы // Вестник кибернетики. 2022. № 1(45). С. 72–82.
8. Ковалева К.А., Яхонтова И.М. Теория исследования и разработки методов и моделей прогнозирования временных рядов с приращением в страховании // Новые технологии. 2019. № 4. С. 239–248.
9. Зюсько К.Д. Прогноз спроса на товар с помощью нейронных сетей в условиях меняющейся размерности входных данных // Экономика и качество систем связи. 2020. № 1 (15). С. 36–41.

10. Луценко Е.В. Применение автоматизированного системно-когнитивного анализа банковских баз данных по операциям с кредитными картами для количественной оценки риска мошенничества // Научный журнал КубГАУ. 2021. № 172. С. 82–172.
11. Кузьмин В.Н., Менисов А.Б. Исследование путей и способов повышения результативности выявления компьютерных атак на объекты критической информационной инфраструктуры // Информационно-управляющие системы. 2022. № 4. С. 29–43.
12. Leichtnam L., Totel E., Prigent N., Me L. Sec2graph: Network attack detection based on novelty detection on graph structured data // Detection of Intrusions and Malware, and Vulnerability Assessment: 17th International Conference, DIMVA. Springer International Publishing, 2020. pp. 238–258.
13. Жукова Н.А. Онтологические модели трансформации данных о состоянии технических объектов // Онтология проектирования. 2019. Т. 9. № 3(33). С. 345–360.
14. Nguyen D., Luo W., Nguyen T., Venkatesh S., Phung D. Sqn2Vec: Learning Sequence Representation via Sequential Patterns with a Gap Constraint. Machine Learning and Knowledge Discovery in Databases. Proceedings of the European Conference, ECML PKDD (Part II). 2019. pp. 569–584.
15. Fradkin D., Morchen F. Mining sequential patterns for classification. Knowledge and Information Systems. 2015. № 45 (3). pp. 731–749.
16. Привалов А.Н., Смирнов В.А. Метод нечеткого сравнения строк для обнаружения фейковых сайтов // Известия ТулГУ. Технические науки. 2022. № 2. С. 184–191.
17. Blanchard P. Sequence Analysis. Encyclopedia of Research Methods. London: Sage Publications Ltd. 2020. URL: https://www.researchgate.net/publication/342232021_Sequence_Analysis (дата обращения: 15.05.2024).
18. Vanasse A., Courteau J., Courteau M., Benigeri M., Chiu Y.M., Dufour I., Couillard S., Larivée P., Hudon C. Healthcare utilization after a first hospitalization for COPD: a new approach of State Sequence Analysis based on the '6W' multidimensional model of care trajectories. BMC Health Serv. Res. 2020. vol. 20(1). DOI: 10.1186/s12913-020-5030-0.
19. Su H., Liu S., Zheng B., Zhou X., Zheng K. A survey of trajectory distance measures and performance evaluation. The VLDB Journal. 2020. № 29. pp. 3–32.
20. Калихман И.Л., Войтенко М.А. Динамическое программирование в примерах и задачах: Учеб. Пособие. М.: Высш. школа, 1979. 125 с.
21. Коган Д.И. Динамическое программирование и дискретная многокритериальная оптимизация: учебное пособие. Нижний Новгород: Изд-во Нижегородского ун-та, 2004. 150 с.
22. Баширзаде Л.И., Алиев Г.С. Применение динамического программирования для моделирования процессов принятия решений // Архивариус. 2022. № 3 (66). С. 51–55.
23. Винцюк Т.К. Анализ, распознавание и интерпретация речевых сигналов. К.: Наук. думка, 1987. 262 с.
24. Шелепов В.Ю., Дорохин О.А., Засыпкин А.В., Червин Н.А. О некоторых подходах к проблеме компьютерного распознавания устной русской речи // Труды Междунар. конф. «Знание – Диалог – Решение». 1997. Т. 1. С. 234–240.
25. Alshehri M., Coenen F., Dures K. Sub-sequence-based dynamic time warping. Proceedings of the 11th International Conference on Knowledge Discovery and Information Retrieval. 2019. pp. 274–281.

26. Deriso D., Boyd S. A general optimization framework for dynamic time warping // Optimization and Engineering. 2023. vol. 24. pp. 1411–1432.
27. Wang L., Koniusz P. Uncertainty-DTW for Time Series and Sequences. European Conference on Computer Vision (ECCV 2022). Cham: Springer Nature Switzerland. 2022. vol. 13681. pp. 176–195.
28. Bringmann K., Fischer N., Hoog I., Kipouridis E., Kociumaka T., Rotenberg E. Dynamic Time Warping // Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms (SODA). Publisher Society for Industrial and Applied Mathematics. 2024. pp. 208–242.
29. Jain V., Fokow V., Wicht J., Wetzker U. A Dynamic Time Warping Based Method to Synchronize Spectral and Protocol Domains for Troubleshooting Wireless Communication // IEEE Access. 2023. vol. 11. pp. 64668–64678.
30. Козлов А.В., Саввина Г.В., Шелепов В.Ю. Система пофонемного распознавания отдельно произносимых слов // Искусственный интеллект. 2003. № 1. С. 156–165.
31. Дорохина Г.В. Модификация алгоритма DTW для пофонемного распознавания слов // Проблемы искусственного интеллекта. 2015. № 0(1). С. 38–49.
32. Дорохина Г.В. Анализ методов распознавания речевых команд на основе алгоритма DTW // Труды шестого междисциплинарного семинара «Анализ разговорной русской речи» (АР3-2012) (27-28 августа 2012. г. Санкт-Петербург). 2012. С. 29–34.
33. Васильев В.И., Шевченко А.И., Эш С.Н. Принцип редукции в задачах обнаружения закономерностей: Монография. Донецк, 2009. 340 с.
34. Бурибаева А.К., Дорохина Г.В., Ниценко А.В., Шелепов В.Ю. Сегментация и дифонное распознавание речевых сигналов // Труды СПИИРАН. 2014. Т. 31. № 8. С. 20–42.
35. Дорохина Г.В., Павлюкова А.П. Модуль морфологического анализа слов русского языка // Искусственный интеллект. 2004. № 3. С. 636–642.
36. Дорохина Г.В. Патент на изобретение № UA 78806 «Устройство для хранения и поиска строковых величин и способ хранения и поиска строковых величин». собственник: Институт проблем искусственного интеллекта. Промышленная собственность. 2007. опубл. 25.04.2007.
37. Дорохина Г.В., Павлыш В.Н. Способ представления множеств последовательностей // Информатика и кибернетика. 2016. № 1(3). С. 56–64.
38. Дорохина Г.В. Сравнение затрат памяти для метода деревьев цифрового поиска и его усовершенствования // Искусственный интеллект. 2009. № 4. С. 338–343.
39. Финаев В.И., Дорохина Г.В. Применения усовершенствованных деревьев цифрового поиска // Проблемы искусственного интеллекта. 2019. № 4 (15). С. 62–77.
40. Bantay L., Abonyi J. Frequent pattern mining-based log file partition for process mining // Engineering Applications of Artificial Intelligence. 2023. vol. 123. DOI: 10.1016/j.engappai.2023.106221.
41. Xing Z., Pei J., Keogh J. A brief survey on sequence classification // SIGKDD Explor. 2010. vol. 12(1). pp. 40–48.
42. Atar R.H., Bhosale D.S. Pattern Based Sequence Classification // International Journal of Advanced Research in Science, Communication and Technology (IJARSCT). 2023. vol. 3. № 1. pp. 390–396.
43. Lazzari N., Poltronieri A., Presutti V. Classifying sequences by combining context-free grammars and OWL ontologies // European Semantic Web Conference. Cham: Springer Nature Switzerland, 2023. С. 156–173.
44. Crochemore M., Lecroq T, Liu L., Ozsu T. Encyclopedia of Database Systems. Verlag: Springer. 2009. pp. 3179–3182.

Дорохина Галина Владимировна — научный сотрудник, ФГБНУ "Институт проблем искусственного интеллекта". Область научных интересов: представление и обработка множеств последовательностей, динамическое программирование, распознавание образов, анализ текстов, разработка программного обеспечения, инженерия знаний, онтологии. Число научных публикаций — 48. sgv_iai@mail.ru; улица Артёма, 118б, 283048, Донецк, Россия; р.т.: +7(856)311-3424.

Поддержка исследований. Работа выполнена в рамках НИР №Г/Р 123092600030-4.

DOI 10.15622/ia.23.6.8

G. DOROKHINA

PHONEME-BY-PHONEME SPEECH RECOGNITION AS A CLASSIFICATION OF SERIES ON A SET OF SEQUENCES OF ELEMENTS OF COMPLEX OBJECTS USING AN IMPROVED TRIE-TREE

Dorokhina G. **Phoneme-by-Phoneme Speech Recognition as a Classification of Series on a Set of Sequences of Elements of Complex Objects Using an Improved Trie-Tree.**

Abstract. Sequences, including vector sequences, are applicable in any subject domains. Sequences of scalar values or vectors (series) can be produced by higher-order sequences, for example: a series of states, or elements of complex objects. This academic paper is devoted to the application of an improved trie-tree in the classification of series on a set of sequences of elements of complex objects using the dynamic programming method. The implementation areas of dynamic programming have been considered. It has been shown that dynamic programming is adapted to multi-step operations of calculating additive (multiplicative) similarity/difference measures. It is argued that the improved trie-tree is applicable in the problem of classifying a series on a set of sequences of elements of complex objects using such similarity/difference measures. An analysis of hierarchical representations of sets of sequences has been performed. The advantages of the improved trie-tree over traditional representations of other highly branching trees have been described. A formal description of the improved trie-tree has been developed. An explanation has been given to the previously obtained data on a significant speed gain for operations of adding and deleting sequences in the improved trie-tree relative to the use of an array with an index table (24 and 380 times, respectively). The problem of phoneme-by-phoneme recognition of speech commands has been formulated as a problem of classifying series on a set of sequences of elements of complex objects and a method for its solving has been presented. A method for classifying a series on a set of sequences of elements of complex objects using the improved trie-tree is developed. The method has been studied using the example of phoneme-by-phoneme recognition with a hierarchical representation of the dictionary of speech command classes. In this method, recognition of speech commands is executed traversing the improved trie-tree that stores a set of transcriptions of speech commands – sequences of transcription symbols that denote classes of sounds. Numerical studies have shown that classifying a series as sequences of elements of complex objects increases the frequency of correct classification compared to classifying a series on a set of series, and using the improved trie-tree reduces the time spent on classification.

Keywords: trie-tree, sets of sequences, classification of series on a set of sequences of elements of complex objects, dynamical programming, phoneme-by-phoneme recognition of speech commands.

References

1. Wirth N. Algorithms and Data Structures. Oberon version. 2004. 211 p.
2. Knut D.Je. Iskusstvo programmirovanija. Vol. 3: Sortirovka i poisk. [The Art of Computer Programming. Vol. 3: Sorting and Searching]. M.: Vil'jams. 2000. 832 p. (In Russ.).
3. Briandais R. File searching using variable-length keys. Proc. Western Joint Computer Conf. 1959. pp. 295–298.
4. Gusfield D. Algorithms on Strings, Trees, and Sequences – Computer Science and Computational Biology. Davis: University of California, 1997. 556 p.

5. Liao T.F., Bolano D., Brzinsky-Fay C., Cornwell B., Fasang A.E., Helske S., Piccarreta R., Raab M., Ritschard G., Struffolino E., Studer M. Sequence analysis: Its past, present, and future. Social science research. 2022. vol. 107. DOI: 10.1016/j.ssresearch.2022.102772.
6. Mathew S., Peat G., Parry E., Sokhal B.S., Yu D. Applying sequence analysis to uncover 'real-world' clinical pathways from routinely collected data: a systematic review. Journal of Clinical Epidemiology. 2024. vol. 166. DOI: 10.1016/j.jclinepi.2023.111226.
7. Gromov V.A., Mazayshvili K.V., Zaikin P.V., Nikolaev E.N., Beschastnov Yu.N., Zvorykina E.I., Parinov A.A., Neznanov A.A. [Differentiating Chaotic and Regular Time Series for Identification of Arteriovenous Fistula State]. Vestnik kibernetiki – Proceedings in Cybernetics. 2022. no. 1(45). pp. 72–82. (In Russ.).
8. Kovaleva K.A., Yahontova I.M. [Research and development theory methods and models for forecasting time series with insurance increments]. Novye Tehnologii – New technologies. 2019. no. 4(50). pp. 239–248. (In Russ.).
9. Zyus`ko K.D. [Forecasting demand for goods using neural networks in conditions of changing dimensionality of input data]. E`konomika i kachestvo sistem svyazi – Economics and quality of communication systems. 2020. no. 1(15). pp. 36–41. (In Russ.).
10. Lucenko E.V. [Application of automated system-cognitive analysis of bank databases on credit card transactions to quantify the risk of fraud]. Nauchny`j zhurnal KubGAU – Scientific Journal of KubSAU. 2021. vol. 172. pp. 82–172. (In Russ.).
11. Kuz`min V.N., Menisov A.B. Investigation of ways and means to improve the effectiveness of detecting computer attacks on critical information infrastructure facilities. Informacionno-upravlyayushhie sistemy` – Information management systems. 2022. no. 4. pp. 29–43. (In Russ.).
12. Leichtnam L., Totel E., Prigent N., Me L. Sec2graph: Network attack detection based on novelty detection on graph structured data. Detection of Intrusions and Malware, and Vulnerability Assessment: 17th International Conference, DIMVA. Springer International Publishing, 2020. pp. 238–258.
13. Zhukova N.A. [Ontological models of transformation of data on the state of technical objects]. Ontologiya proektirovaniya – Design Ontology. 2019. vol. 9. no. 3(33). pp. 345–360. (In Russ.).
14. Nguyen D., Luo W., Nguyen T., Venkatesh S., Phung D. Sqn2Vec: Learning Sequence Representation via Sequential Patterns with a Gap Constraint. Machine Learning and Knowledge Discovery in Databases. Proceedings of the European Conference, ECML PKDD (Part II). 2019. pp. 569–584.
15. Fradkin D., Morchen F. Mining sequential patterns for classification. Knowledge and Information Systems. 2015. № 45 (3). pp. 731–749.
16. Privalov A.N., Smirnov V.A. [Fuzzy string match method for detecting fake sites]. Izvestiya TulGU. Tehnicheskie nauki – News of the Tula state university. Technical sciences. 2022. no. 2. pp. 184–191. (In Russ.).
17. Blanchard P. Sequence Analysis. Encyclopedia of Research Methods. London: Sage Publications Ltd. 2020. URL: https://www.researchgate.net/publication/342232021_Sequence_Analysis (дата обращения: 15.05.2024).
18. Vanasse A., Courteau J., Courteau M., Benigeri M., Chiu Y.M., Dufour I., Couillard S., Larivée P., Hudon C. Healthcare utilization after a first hospitalization for COPD: a new approach of State Sequence Analysis based on the '6W' multidimensional model of care trajectories. BMC Health Serv. Res. 2020. vol. 20(1). DOI: 10.1186/s12913-020-5030-0.

19. Su H., Liu S., Zheng B., Zhou X., Zheng K. A survey of trajectory distance measures and performance evaluation. The VLDB Journal. 2020. № 29. pp. 3–32.
20. Kalihman I.L., Vojtenko M.A. Dinamicheskoe programmirovanie v primerah i zadachah: Ucheb. Posobie [Dynamic programming in examples and problems: Textbook]. Moskow: Vyssh. shkola, 1979. 125 p. (In Russ.).
21. Коган Д.И. Динамическое программирование и дискретная многокритериальная оптимизация: учебное пособие. Нижний Новгород: Изд-во Нижегородского ун-та, 2004. 150 с.
22. Bashirzade L.I., Aliev G.S. [Application of dynamic programming to modeling decision making processes]. Arhivarius. 2022. no. 3(66). pp. 51–55. (In Russ.).
23. Vintsyuk T.K. Analiz, raspoznavaniye i interpretatsiya rechevykh signalov [Analysis, recognition and interpretation of speech signals]. K.: Nauk. dumka, 1987. 262 p. (In Russ.).
24. Shelepov V.Y., Dorokhin O.A., Zasipkin A.V., Chervin N.A. [On some approaches to the problem of computer speech recognition of spoken Russian] «Znanie – Dialog – Reshenie»: trudy Mezhdunar. konf. [Proceedings of the Intern. Conf. "Knowledge – Dialogue – Solution"]. 1997. vol. 1. pp. 234–240. (In Russ.).
25. Alshehri M., Coenen F., Dures K. Sub-sequence-based dynamic time warping. Proceedings of the 11th International Conference on Knowledge Discovery and Information Retrieval. 2019. pp. 274–281.
26. Deriso D., Boyd S. A general optimization framework for dynamic time warping. Optimization and Engineering. 2023. vol. 24. pp. 1411–1432.
27. Wang L., Koniusz P. Uncertainty-DTW for Time Series and Sequences. European Conference on Computer Vision (ECCV 2022). Cham: Springer Nature Switzerland. 2022. vol. 13681. pp. 176–195.
28. Bringmann K., Fischer N., Hoog I., Kipouridis E., Kociumaka T., Rotenberg E. Dynamic Time Warping. Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms (SODA). Publisher Society for Industrial and Applied Mathematics. 2024. pp. 208–242.
29. Jain V., Fokow V., Wicht J., Wetzker U. A Dynamic Time Warping Based Method to Synchronize Spectral and Protocol Domains for Troubleshooting Wireless Communication. IEEE Access. 2023. vol. 11. pp. 64668–64678.
30. Kozlov A.V., Savvina G.V., Shelepov V.U. [Isolated word recognition system based on phoneme recognition]. Iskusstvennyj intellekt – Artificial Intelligence. 2003. vol. 1. pp. 156–165. (In Russ.).
31. Dorokhina G.V. [Modification of the algorithm DTW for spoken word recognition based on phoneme recognition]. Problemy iskusstvennogo intellekta – Problems of artificial intelligence. 2015. vol. 0(1). pp. 38–49. (In Russ.).
32. Dorokhina G.V. [Analysis of speech command recognition methods based on the DTW algorithm] Trudy` shestogo mezhdisciplinarnogo seminara «Analiz razgovornoj russkoj rechi» (AR3-2012) [Proceedings of the sixth interdisciplinary seminar «Analysis of Spoken Russian Speech» (AR3-2012)]. 2012. pp. 29–34. (In Russ.).
33. Vasil'yev V.I., Shevchenko A.I., Esh S.N. Printsip reduktsii v zadachakh obnaruzheniya zakonomernostey: Monografiya [The principle of reduction in the problems of detecting patterns: Monograph]. Donetsk, 2009. 340 p. (In Russ.).
34. Buribayeva A.K., Dorokhina G.V., Nitsenko A.V., Shelepov V.Ju. [Segmentation and diphone recognition of speech signals]. Trudy SPIIRAN – SPIIRAS Proceedings. 2014. vol. 31. no. 8. pp. 20–42. (In Russ.).
35. Dorokhina G.V., Pavlyukova A.P. [Module of morphological analysis of words of the Russian language]. Iskusstvennyy intellect – Artificial Intelligence. 2004. № 3. pp. 636–642. (In Russ.).

36. Dorokhina G.V. Patent No. UA 78806 “Device for saving and searching for lowercase values and method for saving and searching for lowercase values” Owner: Institute of problems of artificial intelligence Promyshlennaja sobstvennost' [Industrial property]. 25.04.2007. (In Russ.).
37. Dorokhina G.V., Pavlysh V.N. [A method of presenting sets of sequences]. Informatika i kibernetika – Informatics and Cybernetics. 2016. № 1(3). pp. 56–64. (In Russ.).
38. Dorokhina G.V. [Memory expenses comparison for the method of digital search tree and its improvement]. Iskusstvennyj intellekt – Artificial Intelligence. 2009. vol. 4. pp. 338–343. (In Russ.).
39. Finayev V.I., Dorokhina G.V. [Applications of improved digital search trees]. Problemy iskusstvennogo intellekta – Problems of artificial intelligence. 2019. vol. 4 (15). pp. 62–77. (In Russ.).
40. Bantay L., Abonyi J. Frequent pattern mining-based log file partition for process mining. Engineering Applications of Artificial Intelligence. 2023. vol. 123. DOI: 10.1016/j.engappai.2023.106221.
41. Xing Z., Pei J., Keogh J. A brief survey on sequence classification. SIGKDD Explor. 2010. vol. 12(1). pp. 40–48.
42. Atar R.H., Bhosale D.S. Pattern Based Sequence Classification. International Journal of Advanced Research in Science, Communication and Technology (IJARSCT). 2023. vol. 3. № 1. pp. 390–396.
43. Lazzari N., Poltronieri A., Presutti V. Classifying sequences by combining context-free grammars and OWL ontologies. European Semantic Web Conference. Cham: Springer Nature Switzerland, 2023. C. 156–173.
44. Crochemore M., Lecroq T, Liu L., Ozsu T. Encyclopedia of Database Systems. Verlag: Springer. 2009. pp. 3179–3182.

Dorokhina Galina — Research fellow, FSBSI “IPAI. Research interests: representation and processing of sets of sequences, dynamic programming, pattern recognition, text analysis, software development, knowledge engineering, ontology. The number of publications — 48. sgv_iai@mail.ru; 118b, Artem St., 283048, Donetsk, Russia; office phone: +7(856)311-3424.

Acknowledgements. The work was carried out within the research No. G/R 123092600030-4.