OSTIS-2015
(Open Semantic Technologies for Intelligent Systems)

УДК 004.891.2

ЧАСТОТНЫЙ МЕТОД КЛАССИФИКАЦИИ ТЕКСТОВ С ЛЕКСИЧЕСКИМ РАЗБОРОМ СЛОВА

Третьяков Ф.И., Серебряная Л.В.

Белорусский государственный университет информатики и радиоэлектроники, г. Минск, Республика Беларусь

Fiodor.Tretyakov@gmail.com

l_silver@mail.ru

В работе рассмотрен алгоритм выделения стема из слова на русском язык. Приведен новый частотный метод классификации. Предложена функциональная модель программного средства универсальной десятичной классификации. Данные методы и средства позволяют сделать присвоение УДК автоматическим с высокой точностью.

Ключевые слова: классификация; стем; удк; эвристика

Введение

На сегодняшний день существует большое количество неупорядоченной текстовой информации. Поэтому поиск и классификация необходимой информации по ключевому слову является одной из важнейших задач. Особенно остро проблема стоит в сфере науки, потому как исследователю часто приходится изучить множество научных работ, прежде чем найти что-то важное для себя. Иногда можно, только взглянув на работу, определить ее тематику, а бывает, что приходится прочитать большую часть текста, чтобы понять его смысл.

Чтобы сразу было известно, к какой области знаний относится научная работа, была придумана универсальная десятичная классификация (УДК). Она является обязательным атрибутом любой печатной или выложенной в электронном виде научной работы. С помощью УДК выполняется классификация информации, необходимая во всем мире для систематизации произведений науки, литературы и искусства, периодической печати, различных видов документов и организации картотек [1].

В настоящее время УДК назначается вручную на основе специальных справочников библиотекарями или специально обученным персоналом. Данная работа посвящена методам и средствам, позволяющим автоматически присваивать работе УДК, не привлекая к этому человека. Поэтому цель работы можно определить как автоматизация универсальной десятичной классификации.

Поставленная задача сводится к тому, что для каждого текста, входящего в множество из n текстов, определить категорию m из УДК.

Предмет исследования работы – универсальная десятичная классификация текстов.

1. Подходы к решению задачи

Существует множество способов решить названную задачу. Прежде всего выбор подхода зависит от количества исходных данных. Если имеется набор текстов-образцов и категорий, то речь идет о контролируемом обучении и классификации. Затем необходимо определить решающее правило и разделяющую функцию, с помощью которых будет выполняться классификация «незнакомых» текстов. Сами тексты обрабатываются и из них выделяются метрики, которые подставляются в качестве параметра в разделяющую функцию, в результате чего определяется принадлежность текстов к одному из классов.

Одним из существенных факторов, влияющих на выбор класса, является язык, на котором написан текст [2]. Настоящая работа посвящена русскоязычным текстам. Поэтому для построения алгоритма классификации будут использоваться особенности русского языка.

Одним из самых популярных способов классификации является поиск по полному

совпадения. Однако он может давать коллизии. К примеру, пользователь задает слово для поиска «кошка». Поэтому в тексте, либо совокупности текстов будут выделяться слова, которые состоят из «кошка..» и любого продолжения этого слова. Такой поиск самый примитивный и совершенно очевидно, что он отсекает такие результаты, которые могли бы быть полезны для пользователя. Например, «кошачий», «кот». Данный способ является самым быстрым, но при этом максимально неточным. Поскольку созданное программное средство сначала не будет работать в режиме реального времени, и современные компьютеры имеют аппаратное обеспечение, способное реализовывать с приемлемым быстродействием «тяжелые» алгоритмы, то необходимо модифицировать алгоритм поиска по полному совпадению с целью повышения его точности [3].

В русском языке лексемы имеют сложные и разнообразные структуры, что существенно затрудняет процедуру классификации текстов. Однокоренные слова могут иметь различные окончания, суффиксы и приставки, которые не должны влиять на результат классификации. Однако при проверке формального совпадения однокоренных лексем и получении отрицательного результата сравнения классификация текстов, построенная на основе неточных результатов сравнения, оказывается неверной. Поэтому для анализа русского языка в качестве разделяющей необходимо выбрать функцию, оперирующую только частью слова и выдающую ответ на его основе. Такой частью может быть корень слова, но он игнорирует специфику русского языка, например, беглые гласные. Поэтому для частотной классификации с максимально возможной точностью его использовать нельзя [2]. Частью слова, которая будет служить его смыслом для классификации, является основа слова [1].

2. Стемминг как часть классификатора

Стемминг – один из способов выделения определенной части слова. Это процесс нахождения основы слова заданной лексемы. Основа не всегда совпадает с морфологическим корнем слова [1]. Задача нахождения основы слова представляет собой давнюю проблему в области компьютерных наук. Первая публикация на заданную тему датируется 1968 годом. Стемминг применяется в поисковых системах для расширения поискового запроса пользователя и является частью процесса нормализации текста. На сегодняшний день созданы различные реализации алгоритмов стемминга. Они применяются для решения различных задач интеллектуальной обработки текстовой информацией.

Для решения задачи классификации используется специальный алгоритм стемминга под названием стеммер [3]. Он может выделять значимую часть слова (стем). Однако стеммер может допускать ошибки, которые классифицируются следующим образом.

Ошибки стемминга 1-го рода. Стем дает слишком большое обобщение и поэтому сопоставляется с грамматическими формами более чем одной словарной статьи. Это самая многочисленная группа ошибок стемминга. К примеру, если при стемминг вами даст вам, то в дальнейшем поиск текста даст совпадение с вампир. В русском языке мжет быть весьма трудно полностью устранить данные ошибки. Например, глагол пасть при спряжении дает формы пади и пал. В результате стемминг дает па, и это очень большое расширение при поиске. Впрочем, ошибки такого типа могут рассматриваться и как способ включить в поиск однокоренные слова. В примере с кошкой это могут быть формы прилагательного кошачий. Компенсация ошибок первого рода успешно выполняется либо введением списка стоп-слов, либо более качественно – лемматизатором или флексером.

Ошибки стемминга 2-го рода. Усечение формы дает слишком длинный стем, которые не сопоставляется с некоторыми грамматическими формами этого же слова. К таким ошибкам приводит стремление разработчика стеммера найти компромисс с ошибками 1-го рода в случае, когда при словоизменении меняется основа слова. Такие слова есть даже в крайне регулярном в плане словоизменения английском языке. Например, группа неправильных глаголов. В русском языке случаи изменения основы даже не является основанием для отнесения слова к группе неправильных, настолько часто это явление. В качестве примера, на котором обычно спотыкаются многие реализации стеммера, можно взять слова кошка и пачка, которые имеют формы кошек и пачек. Обычно стеммеры выполняют в этих случаях усечение до кошк и птичк, которые несопоставимы с формами родительного и винительного падежа множественного числа.

Ошибки стемминга 3-го рода. Стем построить невозможно из-за изменения в корне слова, которое оставляет единственную букву в стеме. Либо модель словоизменения подразумевает использование приставок. Пример для первого случая – глагол впиться, имеющий форму вопьемся. Второй случай возникает в рамках грамматического словаря для сравнительной степени прилагательных и наречий в русском языке. Например, покрасивее как форма прилагательного красивый, или помедленнее как форма наречия медленно.

Для качественного выделения основы слова одного стеммера оказывается недостаточно. Для работы с русским языком можно использовать два дополнительных модуля грамматического словаря: лемматизатор и флексер (склонение и спряжение). С помощью лемматизатора слова приводят к базовой форме, что выполняется после обработки лексемы стемом. Флексер умеет выдавать все грамматические формы слова на основе базовой.

Это позволяет улучшить результат, проверяя найденные фрагменты по набору форм ключевого слова.

Среди всех реализаций стеммеров можно выделить два типа:

1. Использующие словарь для выделения части слова;
2. Использующие эвристическую модель [2].

3. Алгоритм выделения стема

Для выделения корня слова был разработан программный модуль, включающий в себя стеммер, флексер и лемматизатор. Стеммер использует эвристическую модель.

Для создания эвристического стеммера необходимы словари окончаний, формы причастий и деепричастий, суффиксов и приставок. По данным словарей и будет эвристически определяться часть речи. Суть алгоритма сводится к определению части речи для слова по его окончанию, используя словари окончаний. Порядок определения задается уникальностью окончания данной части речи. К примеру, окончания причастий невозможно спутать ни с чем другим, поэтому, стемминг начинается именно с них.

Для создания эвристического стеммера необходимы словари окончаний, формы причастий и деепричастий, суффиксов и приставок. По данным словарям и будет эвристически определяться часть речи. Суть алгоритма сводится к определению части речи для слова по его окончанию используя словари окончаний. Порядок определения задается уникальностью окончания данной части речи. К примеру, окончания причастий невозможно спутать ни с чем другим, поэтому, стемминг начинается именно с них.

Далее приведены словари окончаний:

Причастия и деепричастия:

Группа 1: в, вши, вшись.
Группа 2: ив, ивши, ившись, вы, ывши, ывшись.

Окончания группы 1 должны следовать после: а или я.

Прилагательные:

ее, ие, ые, ое, ими, ыми, ей, ий, ый, ой, ем, им, ым,ом, его, ого, ему, ому, их, ых, ую, юю, ая, яя, ою, ею.

Частицы:

Группа 1: ем, нн, вш, ющ, щ.
Группа 2: ивш, ывш, ующ.

Окончания группы 1 должны следовать после: а или я.

Возвратные местоимения:

ся, сь.

Глаголы:

Группа 1: ла, на, ете, йте, ли, й, л, ем, н, ло, но, ет, ют, ны, ть, ешь, нно.
Группа 2: ила, ыла, ена, ейте, уйте, ите, или, ыли, ей, уй, ил, ыл, им, ым, ен, ило, ыло, ено, ят, ует, уют, ит, ыт, ены, ить, ыть, ишь, ую, ю.

Окончания группы 1 должны следовать после: а или я.

Существительные:

а, ев, ов, ие, ье, е, иями, ями, ами, еи, ии, и,ией, ей, ой, ий, й, иям, ям, ием, ем, ам, ом, о, у, ах, иях, ях, ы, ь, ию, ью, ю, ия, ья, я.

Превосходная степень существительного:

ейш, ейше.

Словообразовательные:

ост, ость.

Приставки

Стемминг будет проходить по следующему алгоритму.

1. Происходит поиск окончаний причастий и деепричастий в слове. Если оно найдено, то удаляется и выполняется переход к шагу 3.
2. Осуществляется поиск окончаний прилагательных, глаголов или существительных. Если они найдены, то удаляются.
3. Если слово оканчивается на «и», оно удаляется.
4. С начала слова в нем ищется последовательность: гласная-согласная. Все буквы после этого сочетания будут блоком n. Если ее нет или блок n пустой, переход к шагу 7.
5. Ищется в блоке n блок m. Это блок, который следует после конструкции гласная-согласная. Если его нет, или он пустой, переход к шагу 7.
6. Ищутся в блоке m части слова «ост» и «ость». Если они найдены, то удаляются.
7. Если слово имеет окончание «ейш» или «ейше», то оно удаляется.
8. Если на конце слова найдено удвоенное «н», второе «н» удаляется.
9. Если на конце слова «ь», он удаляется.
10. Удаляется любое окончание, на которое оканчивается блок *n*.
11. Блок *n*+первая буква *m* и будет стемом.

Стеммер позволяет сделать поиск в тексте на русском языке более осмысленным и логичным. Минусами является сложность модуля и пониженная точность.

Примером данного алгоритма служит рисунок 1, где рассматривается слово «поигравший». На блоках сверху показаны шаги алгоритма от 0 (до алгоритма) и до 11 (после алгоритма). Перечеркнутым начертанием выделены буквы, которые будут удалены на текущей итерации. Жирным – найденное соответствие. Увеличенным шрифтом – искомый блок.

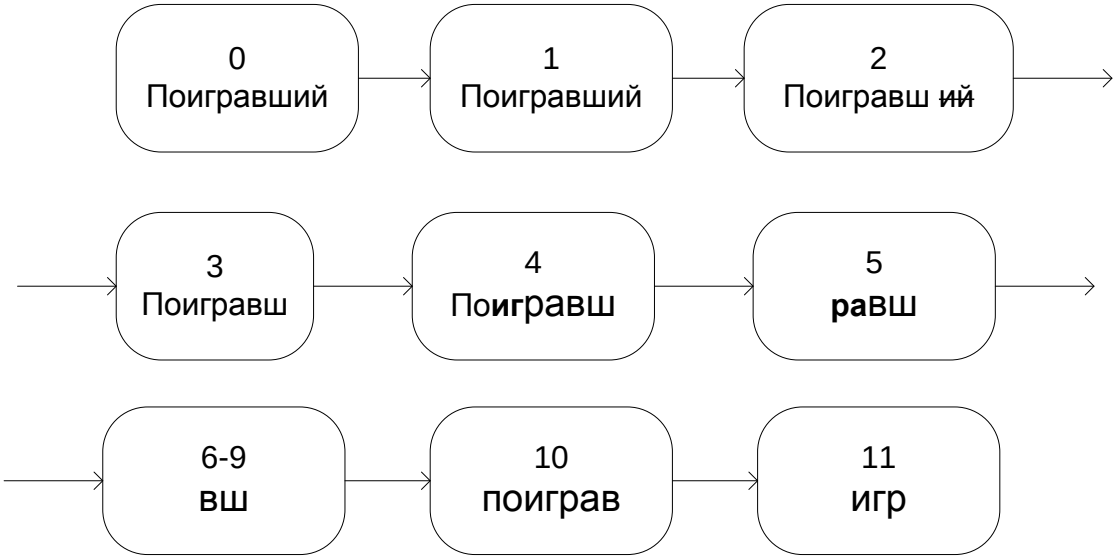

Рисунок 1 – Создание стема

4. Классификация текстов

Следующим шагом будет создание модуля классификации. Рассмотрим алгоритм классификации на основе созданного стеммера.

1. Происходит названий всех категорий с помощью модуля, выделив корни слов и поместив результаты в соответствующий словарь. Каждая строка в нем имеет ключ, является корень слова, а значение в строке – количество всех словоформ по ключу из названия категории.
2. Выполняется шаг 1 для всех , применив его не к названиям текстов, а к ним самим.
3. Для каждого текста находится подходящая категория. Ее номер определяется значения переменной T, вычисленной по следующей формуле:

$$T = \sum_{i=0,\, j=0}^{i<n,\, j<m} a_i \times b_j \,, \qquad (1)$$

где n – размер словаря категории,
m – размер словаря текста,
a_i – слово из словаря категории,
b_k – слово из словаря текста.

4. Происходит выбор категории для текста, где T максимально.

Заключение

С помощью стеммера, флексера и лемматизатора можно классифицировать тексты с высокой точностью. Минусом является сложность архитектуры модуля.

Чтобы продемонстрировать работу данного алгоритма его необходимо встроить в программное средство, оформив в программный модуль. Модуль будет иметь следующую спецификацию:

1. Принимать на входе текст.
2. Выполнять классификацию.
3. Заносить данный текст в базу данных с соответствующим индексом, чтобы учесть данный результат в последующих классификациях.
4. Вывести результат пользователю на экран.

В итоге мы получается систему, которая позволяет присвоить УДК тексту с высокой скоростью, точностью и автоматизированно.

Библиографический список

[Толстых, 2012] Глубинный анализ текста. Из цикла лекций «Современные Internet-технологии» / Толстых В.Л.– М. : Вильямс, 2012.

[Браславский, 2005] Избранные прикладные задачи информатики / Браславский П.Ю.– М. : Вильямс, 2005.

[Duda, etc. 2005] Pattern classification / Duda R.O., Hart P.E., Stork D.G. N. Y. : John Wiley & Sons, 2001.

TEXT CLASSIFICATION FREQUENCY METHOD WITH WORD LEXICAL ANALYSIS

Tretyakov F.I. *, Serebryanaya L.V. *

**Belarusian State University of Informatics and Radioelectronics, Minsk, Republic of Belarus*

Fiodor.Tretyakov@gmail.com

l_silver@mail.ru

In this article there was considered astemming algorithm for Russian language. There was reviewed a new frequency classification method. This article proposes a functional model of a software for Universal Decimal Classification. These methods and software make it possible to do automatic Universal Decimal Classification with high accuracy.