УДК 004.588
ВАК 13.00.02
РИНЦ 14.35.09

М.В. Бирюкова, Д.П. Мамонтов, О.А. Сычев

Определение опечаток в ответе студента на тестовый вопрос при известном правильном ответе

В статье рассматривается метод поиска опечаток в ответах на тестовый вопрос открытого типа. При этом известен один (или несколько) правильных ответов, что дает небольшой словарь корректных слов по сравнению с обычной задачей поиска опечаток в произвольном тексте, и возможность использовать более сложные методы анализа и определять большее количество возможных опечаток, включая пропущенные и лишние разделители. На основе предложенного метода был реализован модуль определения опечаток в ответах студента в вопросе CorrectWriting для СДО Moodle.

Ключевые слова: *алгоритмы определения опечаток, вопросы с открытым ответом.*

DEFINITION OF TYPOS IN ANSWER OF STUDENT IN KNOWN CORRECT ANSWER

The paper describes method of typo detection in the answers for the questions with open answers. In such questions we know one or several correct answers defining relatively small dictionary of correct words contrasting the usual case of looking for typos in arbitrary text. This fact allows using more complex analysis methods and finding more possible typos, such as extra or missing separators. A typo correction module for the CorrectWriting question type (for Moodle LMS) was developed using proposed methods.

Keywords*: algorithms for determining typos, questions with open answer.*

Введение

Образование – приоритетное направление национальной политики России. Дистанционное образование позволяет выбрать удобный, индивидуальный график для каждого обучающегося. Но основной проблемой дистанционного образования является отсутствие контроля над обучением и подсказок в ситуациях, когда студент не может найти решение задачи сам. Поэтому развитие системы тестов для тренировки (с подсказками) и проверки знаний студентов очень важно. Одним из популярных средств реализации таких тестов является система дистанционного образования (СДО) Moodle, которая используется для обучения и в Волгоградском Государственном Техническом университете.

В СДО Moodle реализовано большое количество типов вопросов, при этом реализована возможность расширять ее новыми. Кафедрой ПОАС ВолгГТУ были реализованы два специализированных типа вопроса открытым ответом в текстовой форме с возможностью получения студентом подсказки. Одним из них является CorrectWriting. Этот тип вопроса используется при изучении формальных и естественных языков, в которых достаточно строго определен порядок слов (а точнее – лексем: базовых смысловых единиц языка [1]). Примерами таких языков могут служить английский язык и язык программирования C++. Данный тип вопроса сочетает использует анализ наибольшей общей подпоследовательности для выявления ошибок в ответе студента [2]. Использование тренировочных тестов с вопросами типа CorrectWriting во время самостоятельной работы студентов дает возможность студентам получать сообщения об ошибках и подсказки по их устранению; а при невозможности решить задачу – итоговую картинку, показывающую как из их ответа сделать правильный. Это экономит время, затраченное преподавателем на разбор со студентами ошибочно выполненных ими заданий и значительно расширяет возможности самостоятельной подготовки студентов [2].

Однако эффективному применению метода анализа последовательности лексем в ответе студента, используемом в типе вопроса CorrectWriting, мешало отсутствие модуля коррекции опечаток. В результате, в случае опечатки в слове студент вместо одного сообщения (о наличии опечатки) получал два – об отсутствии требуемого слова и наличии лишнего (неправильного – с опечаткой).

Поэтому для повышения эффективности обучения необходимо было дать типу вопроса CorrectWriting возможность определять и исправлять опечатки в ответе студента перед анализом последовательности лексем.

Мария Вячеславовна Бирюкова,
магистрант кафедры
«Программное обеспечение
автоматизированных систем»
Тел.: 8 (905) 337-70-47
Эл. почта: mariabirvg@gmail.com
Волгоградский государственный
технический университет
www.vstu.ru

Maria V.Biryukova,
graduate student of the Department
«Automated Systems Software»
Tel.: 8 (905) 337-70-47
E-mail: mariabirvg@gmail.com
Volgograd State Technical University
www.vstu.ru/en

Дмитрий Петрович Мамонтов,
аспирант кафедры
«Программное обеспечение
автоматизированных систем»
Тел.: 8 (927) 523-04-72
Эл. почта: mamontov.dp@gmail.com
Волгоградский государственный
технический университет
www.vstu.ru

Dmitry P. Mamontov,
post-graduate student
of the Department
«Automated Systems Software»
Tel: 8 (927) 523-04-72
E-mail: mamontov.dp@gmail.com
Volgograd State Technical University
www.vstu.ru/en

1. Методы определения опечаток

Опечатка это ошибка, допущенная из-за невнимательности. [3] Опечатка может нарушать порядок букв в слове (сутдент вместо студент), буква может быть пропущена в слове (университт вместо университет), или быть лишней (оцеенка вместо оценка) или же одна буква может быть заменена другой (реутор вместо ректор). В одном слове могут встречаться сразу несколько опечаток (жырна вместо журнал). Иногда опечатка может менять смысл слова (печенье вместо печень).

К опечаткам в последовательностях слов относится склейка слов (одиндень вместо один день) или вставка лишнего разделителя (маги стр вместо магистр).

От 80% до 90% всех опечаток отстоят от оригинала на одно изменение символа. При этом среди всех опечаток, если присутствует одна ошибка, то на удаление символа приходится 8% случаев, на перестановку символа – 4% случаев, на вставку символа – 4% случаев, на замену символа – 80% случаев. В случае двух и более опечаток словах, 80% случаев приходится на транспозицию – перестановку местами двух соседних символов [4].

Для определения опечаток используется методы, основанные на вычислении редакционного расстояния – количества операций, которые необходимо применить к одной строке для преобразования её в другую. К классическим редакционным расстояниям относятся расстояние Хемминга, расстояние Левенштейна, расстояние Дамерау-Левенштейна, взвешенное расстояние, расстояние преобразования [5].

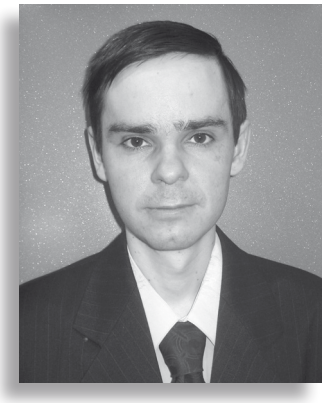
Чаще всего при определении опечаток используется алгоритм Дамерау-Левенштейна [6], который позволяет обнаружить ошибки вставки лишнего символа, удаления символа, замены одного символа на другой и транспозиции символов (наличие последней отличает его от расстояния Левенштейна [7]). При рассмотрении пар соседних лексем его можно использовать и для определения ошибок лишнего или пропущенного разделителя между лексемами. Алгоритм Дамерау-Левенштейна позволяет обнаружить перечисленные виды опечаток на уровне анализа лексем и имеет пригодную для решения задачи алгоритмическую сложность.

Классическая задача поиска опечаток в произвольном тексте требует вычисления редакционных расстояний для большого количества правильных слов, что накладывает очень жесткие ограничения на производительность используемого алгоритма. Особенностью определения опечаток в ответах на тестовые вопросы является небольшой словарь правильных слов, заданный возможными корректными ответами; кроме того, мы знаем, что все слова правильного ответа должны присутствовать в ответе студента без дубликатов. Это позволяет использовать более затратные алгоритмы для поиска вариантов возможных опечаток, в том числе определения отсутствующих или лишних разделителей.

2. Определение опечаток при ограниченном словаре правильных слов

Авторами работы был разработан алгоритм поиска опечаток, особенность которого является наличие ограниченного словаря правильных слов. При наличии нескольких вариантов правильного ответа (введенных преподавателями) вопрос в СДО Moodle сравнивает ответ студента с ними последовательно, выбирая тот, который даст наилучшее совпадение (т.е. наименьшее число ошибок). Поэтому в дальнейшем изложении рассматривается вариант сравнения ответа студента (проверяемая строка) с одним ответом преподавателя (правильная строка).

Перед анализом ответы студента и преподавателя разбиваются на лексемы в соответствии с правилами языка, выбранного автором вопроса. Так как порядок лексем в ответе студента может не совпадать с порядком лексем в ответе преподавателя из-за ошибок их положения, которые будут определяться позже, то для поиска опечаток



Олег Александрович Сычев,
к.т.н., доцент кафедры «Программное обеспечение автоматизированных систем»
Тел.: 8 (905) 434-53-45
Эл. почта: oasychev@gmail.com
Волгоградский государственный технический университет
www.vstu.ru

Oleg A.Sychev,
Candidate of Engineering Science, Associate Professor of the Department «Automated Systems Software»
Tel.: 8 (905) 434-5345
E-mail: oasychev@gmail.com
Volgograd State Technical University
www.vstu.ru/en

необходимо сопоставить каждую лексему из ответа преподавателя с каждой лексемой из ответа студента. При этом должны учитываться варианты ошибок в виде пропущенных и лишний разделителей, в которых лексема одной строки (ответа) сравнивается с парой лексем из другой.

При сопоставлении лексем с вычислением редакционного расстояния между ними формируются пары лексем, которые представляют возможные опечатки. Многие пары изначально следует убрать из рассмотрения ввиду большого редакционного расстояния между входящими в них лексемами, превышающего порог возможной опечатки. Порог этот задается автором вопроса в процентном отношении от длины лексем, так как попытка задать его фиксированным числом ошибок приведет к проблемным ситуациям для слишком длинных, либо слишком коротких слов. Совпадающие лексемы включаются в пары с нулевым редакционным расстоянием.

После получения всех допустимых пар лексем необходимо будет выбрать такой набор пар лексем, который наилучшим образом бы сопоставлял ответ студента и ответ преподавателя. Для этого в наборе лексемы правильного и проверяемого ответов не должны встречаться более одного раза, количество задействованных лексем должно быть максимальным, при этом суммарное редакционное расстояние (при равном количестве лексем) – минимальным.

Для составления оптимального набора был использован обход дерева в глубину [8]. В качестве корня дерева берется пустой набор пар. Каждый уровень дерева добавляет одну пару в набор. Отсечения происходят в связи с тем, что ни одна лексема (из правильного или проверяемого ответа) не должна использоваться в наборе более одного раза. Промежуточные перспективные наборы в процессе поиска сохраняются, заменяясь каждый раз, когда найден набор, покрывающий большее количество лексем или то же количество лексем с меньшим суммарным редакционным расстоянием. Если было найдено несколько наборов с одинаковыми параметрами по обоим критериям (что часто получается, например, если в одном из ответов было две одинаковых лексемы, а в другом только одна из них), то сохраняются все варианты – лучший из них будет определен в дальнейшем по итогам анализа последовательности.

На основе полученного набора (или нескольких) составляется исправленная строка, которая передается следующему модулю для анализа последовательности лексем. Пары с ненулевым редакционным расстоянием, входящие в набор, используются при генерации сообщений об ошибке.

Рассмотрим работу изложенного метода на примере предложения на английском языке (в скобках указан индекс лексемы).

Правильный ответ: the (0) cat (1) ate (2) the (3) mouse (4)

Описания лексем:

the (0) – артикль;

cat (1) – подлежащее;

ate (2) – сказуемое;

the (3) – артикль;

mouse (4) – дополнение.

Проверяемый ответ: at (0) thecat (1) thes (2) mou (3) se (4)

Шаг 1: построение всех возможных пар при пороге соответствия 0.5 (т.е. допустимая доля ошибок составляет 50% длины лексемы) – перечень лексем показан в таблице 1.

Таблица 1

Пары соответствия

Номер пары	Лексемы правильного ответа	Лексемы проверяемого ответа	Редакционное расстояние
0	the (0)	thes (2)	1
1	the cat (0,1)	thecat (1)	1
2	cat (1)	at (0)	1
3	ate (2)	at (0)	1
4	the (3)	thes (2)	1
5	mouse (4)	mou se (3,4)	1

Таблица 2

Варианты наборов (фрагмент)

№	Набор пар лексем	Характеристики набора
1	[0] the (0) – thes (2) – 1 [2] cat (1) – at (0) – 1 [5] mouse (4) – mou se (3,4) – 1	3 пары Ред. расс. = 4 Количество лексем – 7
2	[0] the (0) – thes (2) – 1 [3] ate (2) – at (0) – 1 [5] mouse (4) – mou se (3,4) – 1	3 пары Ред. расс. = 4 Количество лексем – 7
3	[1] the cat (0,1) – thecat (1) – 1 [3] ate (2) – at (0) – 1 [4] the (3) – thes (2) – 1 [5] mouse (4) – mou se (3,4) – 1	4 пары Ред. расс. =5 Количество лексем – 10

Шаг 2: составление наборов пар лексем и выбор наилучшего набора (таблица 2)

Из этих наборов наилучшим, по указанным выше критериям, является третий, так как он покрывает максимальное количество лексем.

Шаг 3: восстановление строки, выделение опечаток

Восстановленная строка (без изменения положений): ate the cat the mouse

Восстановленная строка (с изменением положений): the cat ate the mouse.

Сообщения об ошибках:

1. артикль и подлежащее, возможно, записаны без разделителя;
2. сказуемое, возможно, содержит опечатку;
3. артикль, возможно, содержит опечатку;
4. дополнение, возможно, содержит лишний разделитель;
5. сказуемое перемещено на другое место.

3. Определение опечаток в вопросе CorrectWriting

Рассмотрим примеры работы типа вопроса CorrectWriting при большом количестве опечаток в вопросах, связанных с изучением языка программирования C++. После проверки вопроса студенту выдается перечень сообщений, которые с одной стороны указывают на ошибки, но с другой заставляют задуматься над тем, где они были совершены, после чего он может попытаться их исправить. Если попыток ответа больше не осталось или студент решил прекратить дальнейшие попытки найти правильный ответ, то ему показывается также картинка, объясняющая как из его ответа сделать правильный [2].

В первом случае требуется написать заголовок функции (рисунок 1). Студент совершает несколько опечаток и пропустил разделитель между типом и имением аргумента функции.

Во втором примере требуется написать выражение (рисунок 2). Студент совершает несколько опечаток в одной лексеме. Все опечатки отображаются на картинке корректно и не перекрывают друг друга. Транспозиция показана стрелками.

Рассмотрим попытку студента написать объявление переменной-указателя (рисунок 3). В данном случае студент помимо опечаток совершил пропуск лексемы, которая определяется анализатором последовательности. В сообщениях указываются и опечатки, и другие ошибки.

Для оценки полезности разработанного кода была использована база ответов студентов факультета электроники и вычислительной техники волгоградского государственного технического университета на тестовые вопросы по дисциплине «Основы программирования» в виде фрагментов программ на языке C++. Было проанализировано 2858 попыток ответов на тестовые вопросы от 161 студента. 390 ответов (около 13.5%) содержали опечатки. При этом процент от-

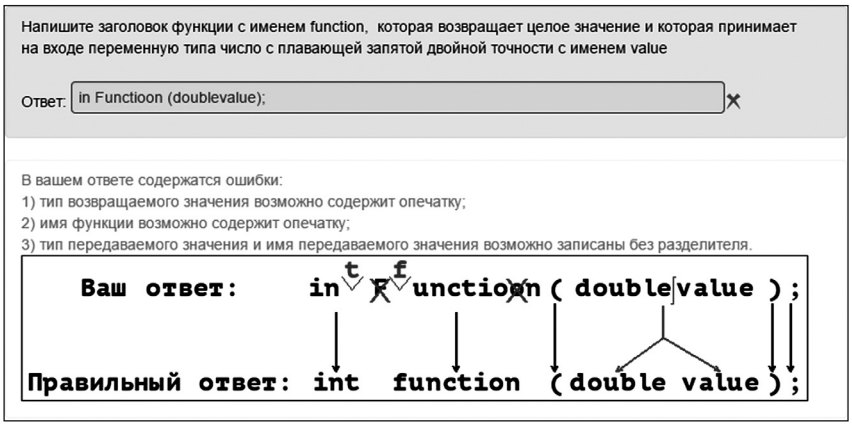


Рис. 1. **Определение опечаток в типе вопроса CorrectWriting: вставка и удаление символов, пропущенный разделитель**

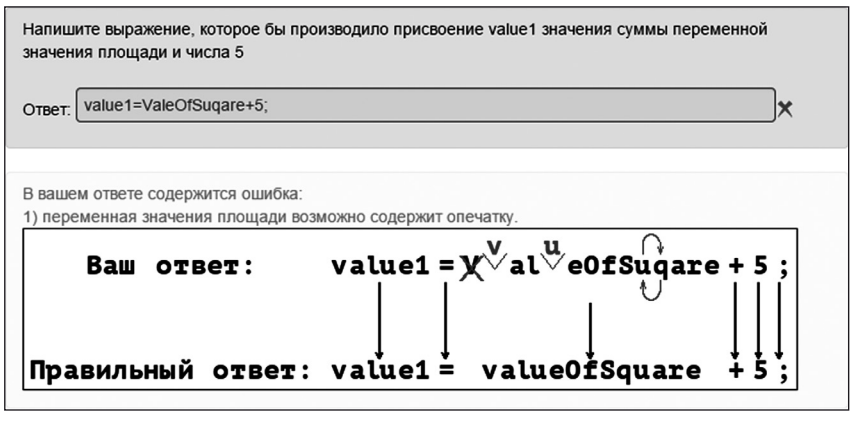


Рис. 2. **Пример работы вопроса CorrectWriting: вставка и удаление символов, транспозиция**

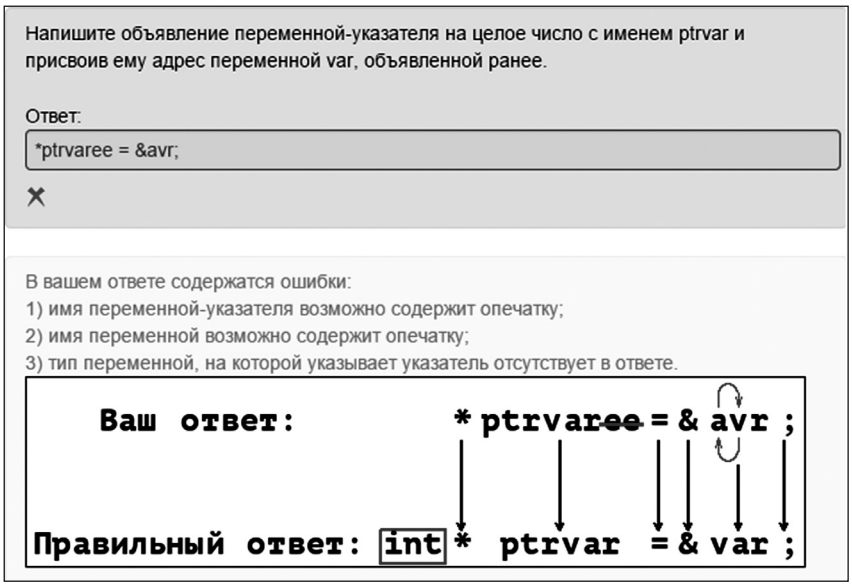


Рис. 3. **Пример работы вопроса CorrectWriting: сочетание различных видов ошибок**

ветов с опечатками слабо зависел от итоговой оценки студентов по дисциплине. Количество опечаток типа «лишний разделитель» и «пропущенный разделитель» было незначительно, что объясняется особенностью языка C++ в котором два слова редко отделяются друг от друга только пробелами: в большинстве случаев между ними стоят скобки, знаки операций или другие знаки препинания. Следует ожидать большего количества опечаток с пропущенным или лишним разделителем в ответах на вопросы на естественных языках.

Заключение

Внедрение модуля определения опечаток в тип вопроса CorrectWriting улучшает возможности оценивания ответов студентов на тестовые вопросы в открытой форме. В зависимости от желания преподавателя вопрос может игнорировать опечатки в ответе студента (если штраф за данный вид ошибок равен 0), либо же выдавать сообщения об ошибке с предположением об опечатке (вместо двух сообщений: об отсутствии правильного написания слова и наличии неправильного).

Однако не все ошибки, определяющиеся по редакционному расстоянию, являются опечатками. Например, в языке C существуют две функции с одинаковым составом параметров и названиями, отличающимися на один символ: strspn и strcspn, при этом смысл выполняемых ими действий противоположен. Аналогично в русском языке приставка «не» к длинному прилагательному вполне может укладываться в порог определения опечатки по редакционному расстоянию, однако ее наличие является смысловой ошибкой. Поэтому авторами планируется дальнейшее совершенствование разработанного модуля для корректного определения специфических для языка смысловых ошибок, похожих на опечатки.

Литература

1. *Ахо А., Сети Р., Ульман Дж.* Компиляторы. Принципы, технологии, инструменты. – М. Вильямс, 2003. – 769 с.
2. *Сычев, О.А.* Автоматическое определение ошибок в порядке расположения лексем в ответах на вопросы с открытым ответом в СДО Moodle / О.А Сычев, Д.П. Мамонтов // Открытое образование. – 2014. – № 2. – С. 79–88.
3. *Андреев А.М.* Автоматизация обнаружения и исправления опечаток в названиях географических объектов для системы семантического контроля документов электронной библиотеки / А.М. Андреев, Д.В. Березкин, А.С. Нечкин, К.В. Симаков, Ю.Л. Шаров // НПЦ «Интелек плюс». – 2007. – № 25.
4. *Карпенко, М.П.* Некоторые методы очистки словаря запросов поиска / М. П. Карпенко, С. В. Протасов // Материалы 17-ой международной конференции по компьютерной лингвистике и интеллектуальным технологиям «Диалог 2011», Наро-Фоминск, 25–29 мая 2011 г.
5. *Гасфилд, Д.* Строки, деревья и последовательности в алгоритмах. Информатика и вычислительная биология. Невский Диалект БВХ-Петербург, 2003, с. 654.
6. *Damerau F.J.* A technique for computer detection and correction of spelling errors // Communications of the ACM, 1964, Vol. 7, Issue 3, p. 171–176.
7. *Levenshtein V.I.* Binary codes capable of correcting deletions, insertions, and revasals // Soviet Physics Doklady: Proceedings of the Academy of Sciences of USSR, Physics section, 1996, Vol. 10.
8. *Касьянов В.Н.* Графы в программировании: обработка, визуализация и применения. / В.Н. Касьянов, В.А. Евстигнеев. – СПб.: БХВ-Петербург, 2003. – 1104 с.