Адаптация модели распознавания речи Google Cloud Speech для упрощения редактирования исходного кода программ для ЭВМ с мобильных устройств

Ю.А. Орлова, А.С. Дмитриев, Д.В. Колчева

Волгоградский государственный технический университет

Аннотация: Рассматривается подход для анализа речи, позволяющий использовать инструменты платформы Google Cloud Speech для редактирования исходного кода программ для ЭВМ, комбинирующий обработку звука, редакционные расстояния и таблицы замены. Данный подход может быть использован для редактирования исходного кода программ для ЭВМ с использованием мобильных устройств. Рассмотрен прототип приложения для редактирования исходного кода программ для ЭВМ, использующий данный подход и поддерживающий интеграцию с платформой GitHub и популярным средством обмена мгновенными сообщениями Telegram.
Ключевые слова: распознавание речи, мобильные устройства, анализ исходного кода, формальные языки, редакционные расстояния.

Введение

В последнее время, в связи с пандемией COVID-19 и другими социальными факторами разработчики приложений на языках высокого уровня очень часто работают дистанционно. При этом условия, в которых находятся сотрудники могут быть удобными, но первоначально не подходящими для рабочего процесса [1]. Вместе с тем отмечается рост применения мобильных устройств для выполнения незначительных текущих задач разработки и поддержки программного обеспечения.

Однако на мобильных устройствах ввод и внесение изменений в исходный код программы могут быть затруднены, так как полноценная клавиатура бывает недоступна или недостаточно эргономична, вместе с тем не исключено, что внесение правок потребуется и в таких условиях, а именно, в случае аварии на главном сервере, обслуживающем разрабатываемую программу.

Для уменьшения времени внесения изменений, когда работа осуществляется в такого рода условиях, применяются различные средства, обладающие своими преимуществами и недостатками. К примеру, в

операционной системе (ОС) Android инструмент ввода представляет экранную клавиатуру, набор на которой осуществляется путём нажатия виртуальных кнопок на дисплее. Этот метод удобен, но при низком разрешении экрана, малом его размере или плохом состоянии сенсора ввод с такой клавиатуры становится затруднителен, возникают ошибки и опечатки.

Ещё один из способов решения рассматриваемой проблемы - использование технологий распознавания речи. Достоинством такого подхода является возможность ввода текста при наличии в помещении вибрации или отсутствии клавиатуры как таковой. Некоторые исследования показывают, что в определённых случаях распознавание речи позволяет достичь большей скорости набора, чем при использовании других методов. Особенно хорошо это применимо к вводу коротких команд (статья Lewis K., Pettey M.) [2].

Встроенные средства распознавания речи существуют и в ОС Android, тем не менее, они несовершенны и не адаптированы для ввода исходного кода программы, особенно, в области спецсимволов, знаков препинания или же ключевых слов. Рассмотрим следующий пример. Пусть пользователь произносит слово «звёздочка», при использовании распознавания речи система может идентифицировать это как «звёздочка» или «*», в большинстве случаев отдаётся предпочтение первому варианту [3]. Это отражено и в используемых средах разработки для данной операционной системы.

Сто́ит учесть, что для операционной системы Android существует множество сред разработки — программ, позволяющих редактирование исходного кода. Достаточно мощными среди них являются Pydroid 3 [4] и air.JavaEditor [5]. Как первая, так и вторая позволяют редактировать файлы исходного кода для программ на языках программирования Python и Java. Приложения могут частично проверять синтаксис, а также подсвечивать

исходный код. Во втором приложении есть собственная клавиатура, которая позволяет немного упростить ввод, а также реализована функция автодополнения, но в нём отсутствует возможность редактирования кода посредством голосовых команд. Таким образом, возникшую проблему нельзя решить, используя современные средства для разработки, хотя технологию распознавания речи используют при внесении незначительных правок в исходный код программы при отсутствии полноценной клавиатуры, а также когда экранная клавиатура неудобна для внесения правок.

Предлагаемый подход для упрощения задачи внесения правок в исходный код программ с использованием мобильных устройств

Среди сервисов для распознавания речи наиболее популярен Google Cloud Speech-to-Text [6,7]. Данный инструмент позволяет осуществлять преобразование речи в текст для множества естественных языков и, соответственно, может быть использован для упрощения задачи внесения правок в исходный код программ. Однако, у него есть следующие проблемы: например, он очень чувствителен к произношению английского языка (так, некоторые слова в зависимости от тона произношения могут распознаваться некорректно), а также общие проблемы при распознавании речи, которые были рассмотрены выше. В качестве примера рассмотрим обычное выражение на языке Python: "while (item <total_items)". В зависимости от тона и интонаций произносящего, сервис может распознать фразу как «I’ll left brain item lesser total underscore items right brain», что является некорректным.

Для решения этой проблемы был разработан подход, который комбинирует таблицы замены, фиксированный набор правил и редакционные расстояния для исправления ошибок при распознавании речи [8,9]. В данном подходе таблицы замены используются для корректировки известных вариантов некорректного распознавания, фиксированный набор правил – для

комбинирования распознанных слов в условиях побуквенного и посимвольного ввода, который чаще всего требуется при наборе идентификаторов (базовых единиц языка программирования, описывающих имена переменных, методов и других). Для распознавания сложных случаев используется динамический поиск с применением редакционных расстояний по списку идентификаторов исходного кода, в который вносятся изменения. Общий ход работы подхода показан на UML-диаграмме активности на рис. 1.

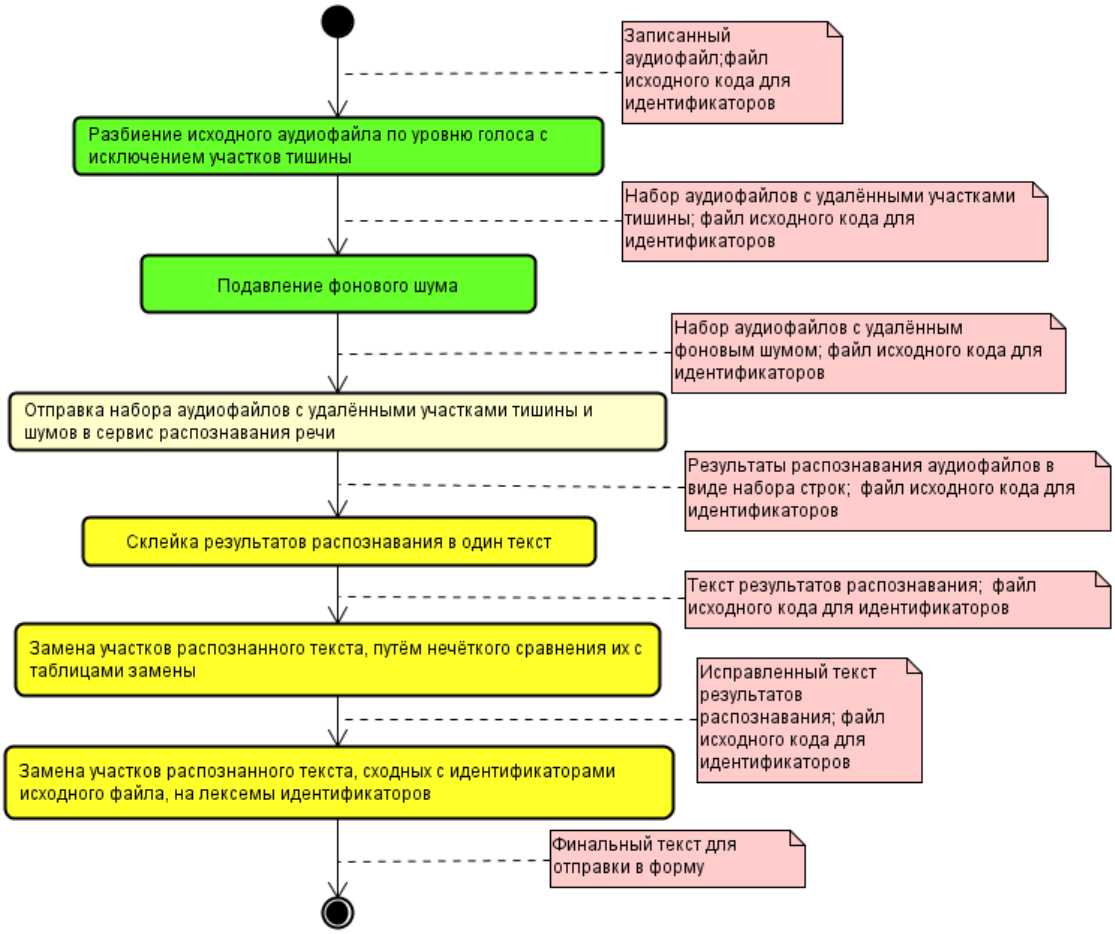

Рис. 1. – Общий ход работы предлагаемого подхода

Помимо уже описанных выше этапов, предварительно, в рамках подхода, осуществляется разбиение исходного аудиофайла по уровню голоса с удалением участков тишины и подавление фонового шума для исключения возможности зависимого распознавания от контекста и улучшения результатов. Для этого могут быть использованы уже готовые решения,

используемые в ряде аудиоприложений и библиотек для языков программирования.

Также сто́ит отметить, что таблиц для замены может быть несколько, исходя из грамматики распознаваемого языка. Модель таблиц можно описать в виде **T = {F, T, REGISTRY, FIRST}**, где:

F – множество исходных лексем для замены при разборе результатов распознавания речи;

T – множество конечных лексем для замены;

REGISTRY – множество флагов, указывающих на то, учитывается ли регистр при замене;

FIRST – множество флагов, указывающих на то, заменяется ли только лексема в начале;

Для сравнения с таблицей при анализе используется последовательное сопоставление двух строк с применением расстояния Дамерау — Левенштейна.

Также реализована возможность добавления пользовательских таблиц замены по типу расширяемого речевого словаря, реализованного в [10].

Сто́ит отметить, что описанный выше подход возможно применять для любых технологий распознавания речи, а не только сервиса Google Cloud Speech-to-Text.

Прототип приложения для упрощения задачи внесения правок в исходный код программ с использованием мобильных устройств

Для демонстрации указанного подхода было разработано веб-приложение Reviewgram на основе веб-клиента популярного мессенджера Telegram – Webogram [11]. Данное приложение предназначено для внесения правок в исходный код программ, размещённых на платформе GitHub, используя мобильные устройства. Эта платформа была выбрана в силу её довольно большой популярности [12].

Общий ход работы приложения показан в виде UML-диаграмме активности на рис. 2.

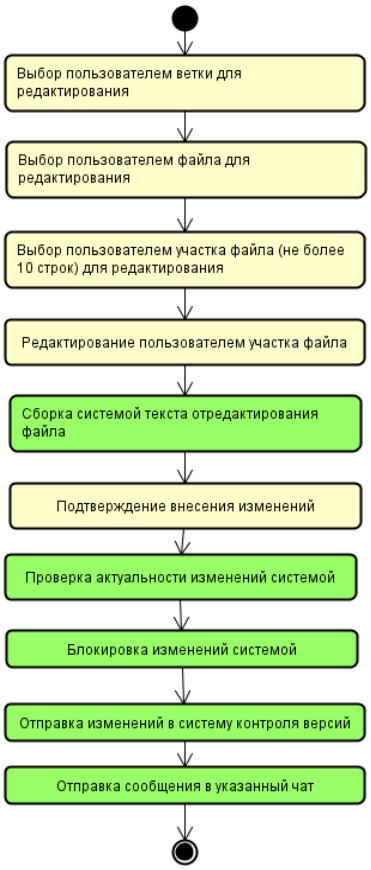

Рис. 2. – Общий ход работы прототипа приложения

В приложении данный подход используется для выбора файла (показано на рис.3).

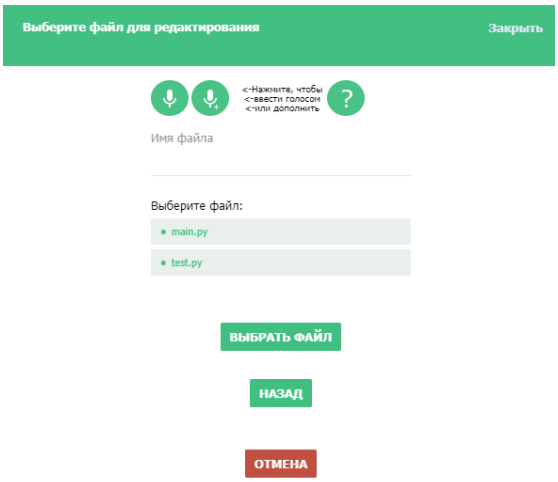

Рис. 3. – Общий ход работы прототипа приложения

После того, как пользователь выбрал файл, ему предлагается выделить промежуток строк для дальнейшего редактирования. Сам интервал показан выделением номеров строк на рис. 4. Ограничение по объёму редактируемого промежутка возникает ввиду неудобства редактирования большого количества строк с мобильных устройств.

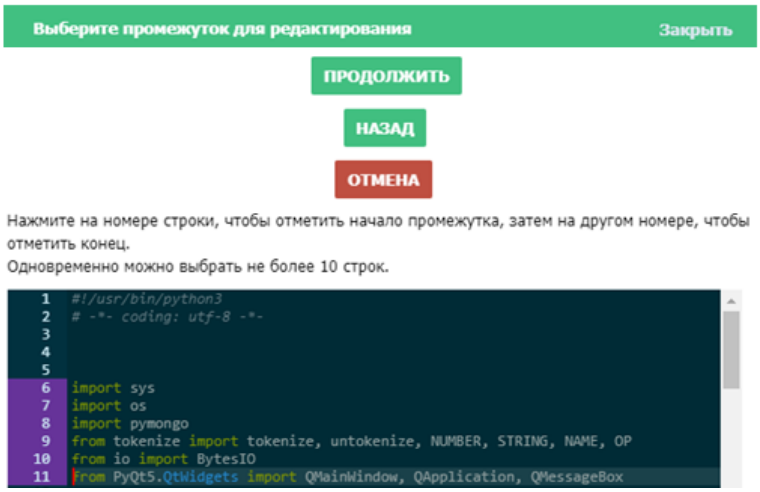

Рис 4. Интерфейс выбора промежутка

Также распознавание речи используется и на этапе ввода команды редактирования текста. Здесь пользователю предлагается набрать текст для

вставки или замены в исходном коде программы, показанном на рис.5.

5.

Рис 5. Интерфейс редактирования файла

После редактирования файла пользователю предлагается подтвердить свои действия. При этом производится проверка на одновременное редактирование одного файла, применяются изменения и отправляется сообщение в соответствующий чат, которое должно оповестить остальную команду, при условии командной разработки, о том, что были внесены изменения.

При распознавании речи приложение позволяет выбрать, необходима ли замена старого текста новым или дополнение предыдущих результатов вновь распознанным текстом. Сто́ит отметить, что данный подход в приложении используется не сам по себе, а вместе с рекомендациями по разработке мобильных интерфейсов [13] и с учётом ограничений, указанных выше. Это было сделано для проектирования пользовательского интерфейса,

в частности, размера кнопок, шрифта, а также ограничения по количеству одновременно редактируемых строк.

Заключение

Использование данного подхода позволяет ускорить процесс внесения мелких правок в исходный код программ на языках программирования высокого уровня. Однако, в процессе реализации, у этого подхода обнаружились недостатки – требовательность к ресурсам для подавления шума и разбиения по словам, что решается выделением достаточных для этого ресурсов и более совершенными алгоритмами, а также трудоёмкость по составлению таблиц замены, которая для языка Python составила несколько дней. Однако, данные недостатки не являются сильно критичными, поэтому авторами планируется дальнейшее совершенствование разработанного подхода и приложения.

Литература

1. Стребков Д.О., Шевчук А.В., Спирина М.О. Самостоятельная занятость на рынке удалённой работы: распространение инновационной трудовой практики // Мониторинг общественного мнения: Экономические и социальные перемены № 6. 2016. - с. 89—106.

2. Lewis K., Pettey M., Shneiderman B. Speech-Activated versus Mouse-Activated Commands for Word Processing Applications: An Empirical Evaluation// Int. J. Man-Machine Studies, №39. - 1993. – pp. 667-687.

3. Орлова Ю.А., Дмитриев А.С., Колчева Д.В. Автоматизация исправления ошибок при редактировании исходного кода программ для ЭВМ с мобильных устройств через использование технологий распознавания речи // Инновационные технологии в обучении и производстве: материалы XIV всерос. заочн. науч.-практ. конф. (г. Камышин, 15 ноября 2019 г.). В 3 т. Т. 2.

/ под общ. ред. И. В. Степанченко; ВолгГТУ, КТИ (филиал) ВолгГТУ. - Волгоград, 2019. - С. 129-133.

4. Pydroid 3 - IDE for Python 3 // Google Play. URL: play.google.com/store/apps/details?id=ru.iiec.pydroid3&hl=ru&gl=US (дата обращения: 27.12.2020).

5. Приложение air.JavaEditor // Google Play. URL: play.google.com/store/apps/details?id=air.JavaEditor&hl=ru&gl=US (дата обращения: 27.12.2020).

6. Kim J., Lu C., Calvo R., McCabe K. L. A Comparison of Online Automatic Speech Recognition Systems and the Nonverbal Responses to Unintelligible. 2019. 13 p.

7. Glasser A. Automatic Speech Recognition Services: Deaf and Hard-of-Hearing Usability // Extended Abstracts of the 2019 CHI Conference. – 2019. – pp. 1-6

8. Дмитриев, А.С, Орлова Ю. А. Автоматизация идентификации естественно-языковых описаний категорий времени и пространства в тексте // Интегрированные модели и мягкие вычисления в искусственном интеллекте : сб. науч. тр. VII-й междунар. науч.-практ. конф. В 3 т. Т. 2 / Рос. ассоциация искусств. интеллекта, Рос. ассоциация нечётких систем и мягких вычислений, МГТУ им. Н.Э. Баумана. - М., 2013. - С. 508-517.

9. Сычев О.А., Мамонтов Д.П. Автоматическое определение ошибок в порядке расположения лексем в ответах на вопросы с открытым ответом в СДО Moodle // Открытое образование. - 2014. - № 2. - С. 79-88.

10. Балакшин П.В. Определение необходимого размера словаря для систем автоматического распознавания речи телефонных служб поддержки клиентов // Инженерный вестник Дона, 2015, №4. URL: ivdon.ru/ru/magazine/archive/n4y2015/3382 (дата обращения: 27.12.2020).

11. Webogram // Github URL: github.com/zhukov/webogram (дата обращения: 27.12.2020).

12. Borges H., Hora A., Valente. M. T. Understanding the Factors That Impact the Popularity of GitHub Repositories//2016 IEEE International Conference on Software Maintenance and Evolution (ICSME). – Raleigh, 2017.

13. Компаниец В.С., Лызь А.Е. Эргодизайн пользовательского интерфейса: методы юзабилити исследований // Инженерный вестник Дона, 2017, №3. URL: ivdon.ru/ru/magazine/archive/n3y2017/4333 (дата обращения: 27.12.2020).

References

1. Strebkov D. O., Shevchuk A. V., Spirina M.O. Ekonomicheskie i social'nye peremeny [Monitoring of public opinion: Economical and social changes] № 6. 2016. pp. 89-106.

2. Lewis K., Pettey M., Shneiderman B. Int. J. Man-Machine Studies, №39. 1993. pp. 667-687.

3. Orlova, Y.A., Dmitriev A.S., Kolcheva D.V. Innovacionnye tehnologii v obuchenii i proizvodstve: materialy XIV vseros. zaochn. nauch.-prakt. konf. (g. Kamyshin, 15 nojabrja 2019 g.). V 3 t. T. 2. Pod obshh. red. I. V. Stepanchenko; VolgGTU, KTI (filial) VolgGTU. Volgograd, 2019. pp. 129-133.

4. Pydroid 3 - IDE for Python 3. Google Play. URL: play.google.com/store/apps/details?id=ru.iiec.pydroid3&hl=ru&gl=US (accessed 27.12.2020)

5. air.JavaEditor Application. Google Play URL: play.google.com/store/apps/details?id=air.JavaEditor&hl=ru&gl=US (accessed 27.12.2020).

6. Kim J., Lu C., Calvo R., McCabe K. L. A Comparison of Online Automatic Speech Recognition Systems and the Nonverbal Responses to Unintelligible Speech. 2019. 13 p.

7. Glasser A. Extended Abstracts of the 2019 CHI Conference. 2019. pp. 1-6.

8. Dmitriev, A. S., Orlova Y. A. sb. nauch. tr. VII-j mezhdunar. nauch.-prakt. konf. . Ros. associaciya iskusstv. intellekta, Ros. associaciya nechyotkih sistem i myagkih vychislenij (VII international science conference. Russian association of artificial intelligence. Russian association of fuzzy systems and soft computing). Moscow, 2013. pp 508-517.

9. Sychev, O.A. Mamontov D.P. Otkrytoe obrazovanie [Open Education]. 2014. № 2. pp. 79-88.

10. Balakshin P.V. Inzhenernyj vestnik Dona, 2015, №4. URL: ivdon.ru/ru/magazine/archive/n4y2015/3382 (accessed 27.12.2020).

11. Webogram. Github. URL: github.com/zhukov/webogram (accessed 27.12.2020).

12. Borges H., Hora A., Valente. M.T. 2016 IEEE International Conference on Software Maintenance and Evolution (ICSME). Raleigh, 2017.

13. Kompaniets V.S., Lyz' A.E. Inzhenernyj vestnik Dona, 2017, №3. URL: ivdon.ru/ru/magazine/archive/n3y2017/4333 (accessed: 27.12.2020).