Разработка самообучающегося бота-собеседника с возможностью расчета арифметического выражения

О.Г. Ведерникова, Н.А. Москат

Ростовский государственный университет путей сообщения

Аннотация: Представлен алгоритм и листинг кода разработанного самообучающегося чат-бота, способного поддерживать беседу с пользователем, переходить в режим обучения в случае отсутствия нужной информации, проводить анализ контекста беседы на «запрещенные» слова. А также реализована дополнительная опция: «Решать арифметические задачи» в случае запроса от пользователя. При реализации последней опции использовался алгоритм рекурсивного спуска на основе трех вложенных рекурсивных функций. Разработана функция для разделения входящей строки на токены. Код чат-бота разработан на языке C++/C# .

Ключевые слова: искусственный интеллект, машинное обучение, голосовой помощник чат-бот, режим обучения, токены, рекурсивный спуск, бинарное дерево.

Предметом данного исследования являются методы машинного обучения и их применение в интернет-приложениях, в частности, разработка ботов – специальных программ, которые выполняют какие-либо действия автоматически, или по определенному алгоритму.

Актуальность работы связана со стремительным ростом популярности технологии искусственного интеллекта (ИИ), обусловленной ростом производительности вычислительной техники [1]. Одним из самых популярных и узнаваемых примеров ИИ является голосовой помощник [2,3]. Известные голосовые помощники: «Яндекс Алиса», «Google Now», «Apple Siri», «Amazon Alexa».

В данной работе представлен самообучающийся чат-бот, с которым пользователь сможет общаться. Пользователь будет вводить вопрос, после чего бот «вспоминает» ответ, если он вспомнил его, то отвечает, иначе пользователь должен будет «научить» бота, как нужно отвечать на этот вопрос. Отличительная черта данного проекта состоит в том, что бот может переходить из режима обучения в режим работы и обратно для общения и поддержания диалога с человеком [4, 5]. В режиме обучения чат-бот

сохраняет в своей базе все пары вопросов и ответов, встретившихся ему в процессе. В режиме работы поддерживается диалог с человеком.

Кроме того, для чат-бота разработан способ анализа контекста диалога на «запрещенные» слова и ухода от ответа на вопросы с запрещенными словами. Причем список таких запрещенных слов может дополняться [6].

Другой важной отличительной чертой проекта является блок для решения арифметических задач, который реализован на основе алгоритма рекурсивного спуска, причем предварительно строка анализируется и разделяется на токены, выполняется синтаксический разбор строки. Код чат-бота разработан на языке C++/C# [7]. Разработан основной алгоритм работы чат-бота, блок-схема которого представлена на рис. 1.

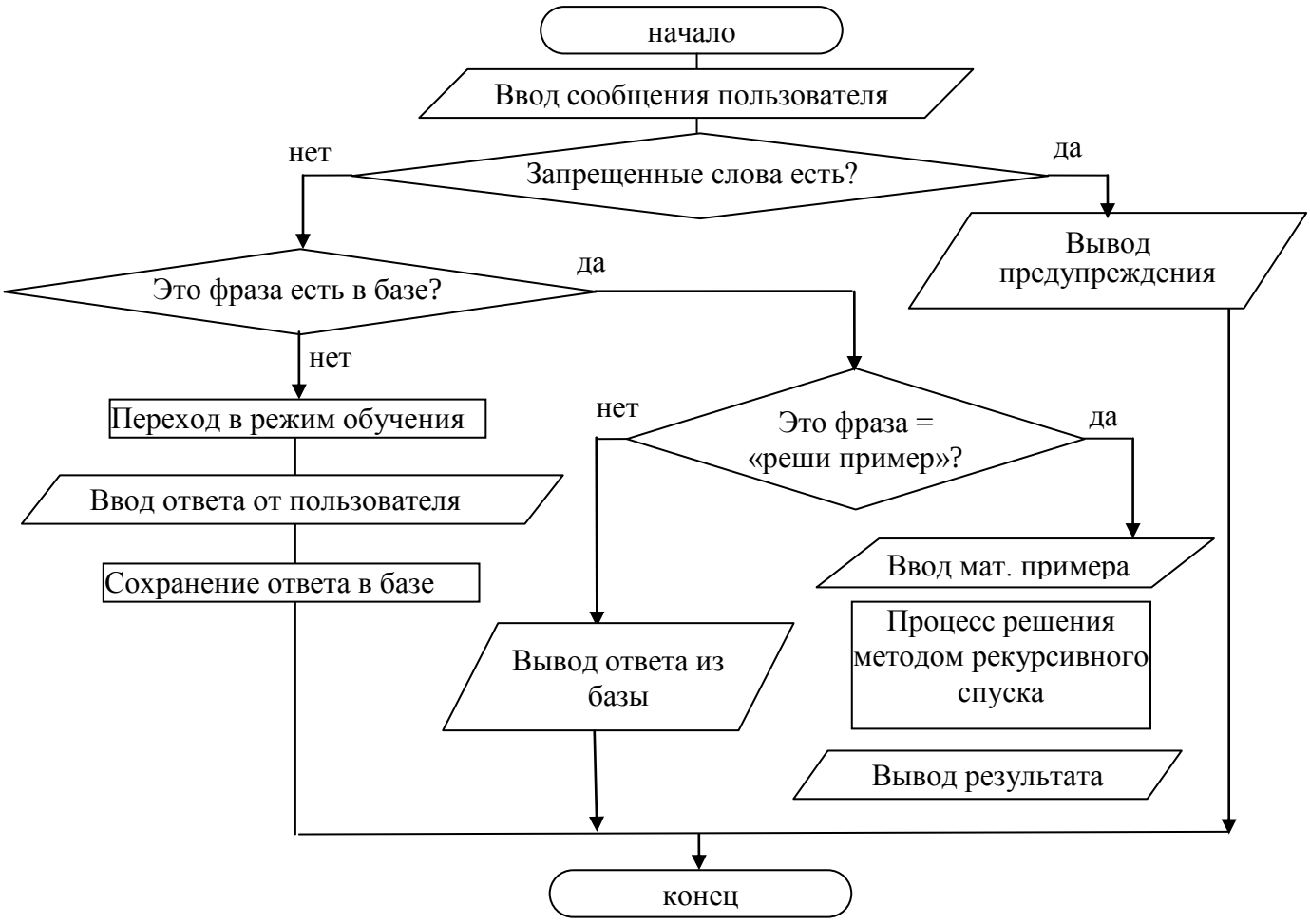

Рис. 1. Блок-схема работы чат-бота

Представленный алгоритм заключен в бесконечный цикл *while (true)*, листинг его приведен на рис. 2. В цикле вводится сообщение от пользователя в переменную *question* и вызывается функция *Answer* для поиска правильного ответа.

```
Console.WriteLine("Ваш вопрос:");
          while (true)
          { string question = Console.ReadLine();
              chatBot.Answer(question);
          }
```

Рис. 2. Основной цикл реализации чат-бота

В функции *Answer* бот переключается между режимами работы и обучения. Листинг этой функции представлен на рис. 3. В свойства класса *ChatBot* включена переменная *bool flag* – переключатель между режимами работы (*flag = true*) и обучения (*flag = false*). А также переменная *string userAnswer*, в которой находится информация об ответе пользователя в режиме обучения, который бот запоминает.

```
public void Answer(string question) //генерация ответа
        { if(flag)
             {  this.question = question;
                string ans = string.Empty;
                question = question.ToLower();
                question = Trim(question, tr.ToCharArray());
                if (Anti_Filthy_Landuage(question))
                  { GetStr("Не ругайся!\nВаш вопрос:");
                    return;
                  }
                string answer = Ans(question);
                //переход в режим обучения
        if (answer == string.Empty)
        {flag = false;
GetStr("Я не знаю ответа, напиши как нужно отвечать: ");}
          //режим работы
    else
  {if(question == "реши пример" && answer == "введите его")
        {GetStr("Мой ответ: " + answer + "\nВаш пример:");
        GetExpression();}
         else
      {GetStr("Мой ответ: " + answer + "\nВаш вопрос:");}
         }          }
      //режим обучения
```

```
                    else
                    {flag = true;
                        question = question.ToLower();
                        userAnswer = question;
if(!(Anti_Filthy_Landuage(userAnswer)||Anti_Filthy_Landuage(question)))
                        {Teach();
                            GetStr("Я запомнил!\n");}
                        else
                        {GetStr("Не ругайся!\n");}
                    GetStr("\nВаш вопрос: ");}
                }
```

Рис. 3. Листинг функции *Answer*, генерирующей ответ бота

При вызове функции *Answer* бот проверяет переменную *flag*. По умолчанию *flag* равен *true*, поэтому бот сначала перейдет в режим работы и вызовет функцию *Ans*, которая возвращает соответствующий ответ на вопрос. После каждого вопроса, бот записывает ответ, поэтому поиск происходит в цикле for с шагом *i*+=2. Если эта функция вернет пустое значение, это будет означать, что бот не знает ответа, после чего сам изменит значение *flag* на *false* и попросит пользователя ввести ответ. Следующий ввод сообщения будет распознаваться ботом, как ответ на вопрос, и бот его запомнит.

Когда бот переходит в режим обучения, вызывается функция *Teach*, которая реализует режим обучения бота (см. рис.4). В переменной *string path* – хранится путь к файлу с ответами. В списке *List<string> sempls* – содержится список всех вопросов и ответов, которые есть в этом файле.

```
                private void Teach()
                {question = question.ToLower();
                    string q = Trim(question, tr.ToCharArray());
                    sempls.Add(q);
                    sempls.Add(userAnswer);
            File.WriteAllLines(path,sempls.ToArray());//сохранение}
```

Рис. 4. Листинг функция *Teach* режим обучения

Для удаления ненужных символов из вопросов используется функция *Trim*. При этом удаляются все символы, представленные в свойстве *tr*. Этот набор символов *tr* можно менять, чтобы сохранялось корректное сообщение [8]. Функция *Trim* применяется в функциях *Answer*, *Ans* и *Teach*.

На рисунке 5 представлено тестирование общения и обучения бота. Как видно из рисунка, на известную ему фразу бот отвечает сразу, а при отправке ему неизвестного сообщения, начинается процесс обучения бота.

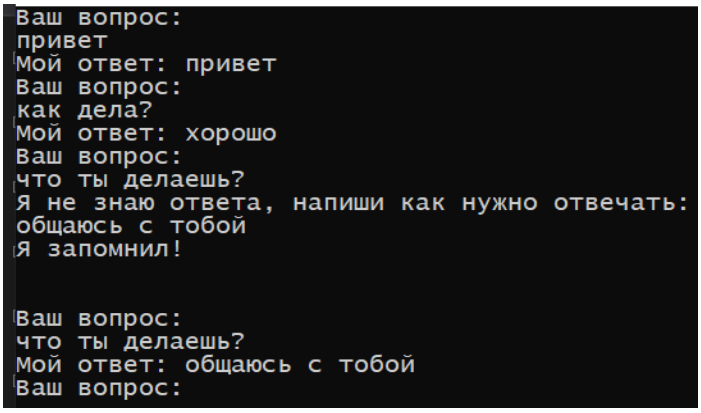

Рисунок 5 – Тестирования общения и обучения бота

В алгоритм чат-бота добавлен фильтр на «запрещенные» слова, который реализован в методе Anti_*Filthy_Landuage*(), проверяющий все вопросы (см. рис. 6.).

```
 private bool Anti_Filthy_Landuage(string input)
{string[] swArray = File.ReadAllLines(pathFL);ьь
string[]words=Trim(input,tr.ToCharArray()).Split
            foreach (string str in swArray)
            {foreach (string str2 in words)
                {if (str == str2) return true;
                 }
            } return false;
        }
```

Рис.6. Метод, проверяющий текст на запрещенные слова

В переменной *string pathFL* храниться путь к файлу с «запрещенными» словами. Из него считываются все слова и сравниваются, если в отправленной пользователем строке найдется такое слово, то функция вернет

false, иначе *true* – запрещенных слов не найдено. Словарь запрещенных слов можно пополнять в соответствующем файле. Тестирование функции *Anti_Filthy_Landuage*() представлено на рис. 7.

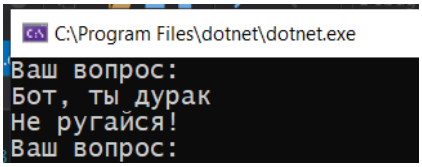

Рис. 7. Тестирование проверки стоки на запрещенные слова.

Добавление возможности боту вычислять арифметические выражения сделает систему еще более насыщенной. В текстовом файле с ответами прописана ключевая фраза «Реши пример», после которой бот ожидает от пользователя ввода арифметического выражения, состоящего только из цифр, скобок и знаков арифметических операций. При реализации блока «Процесс решения» используется метод рекурсивного спуска для синтаксического анализа, схема которого представлена на рис. 8.

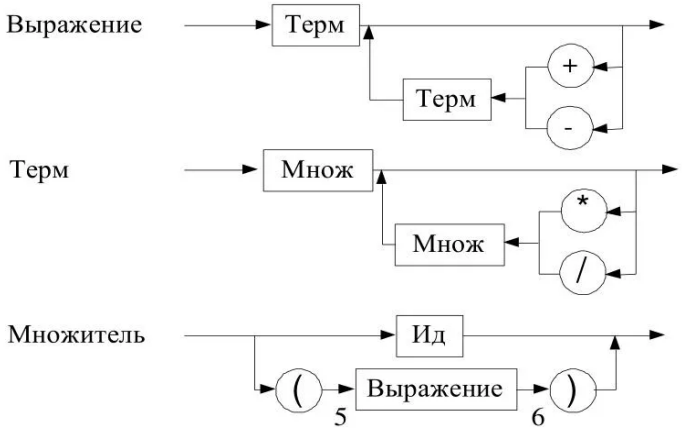

Рис. 8. Схема метода рекурсивного спуска для вычисления

В функции *GetExpression* () реализован метод рекурсивного спуска, которая возвращает значение типа *double*. Сначала проверяется входящая строка – не должно быть знаков, таких, как «№», «?», «!» и т.д. Кроме того, выполняется синтаксический разбор строки, разделяется строка на токены – арифметические знаки и числа, при этом заполняется бинарное дерево.

Разработаны рекурсивные функции E (), T (), F (), служащие для реализации алгоритма спуска, в соответствии со схемой на рис. 8. Тестирование решения арифметического выражение ботом представлено на рис. 9.

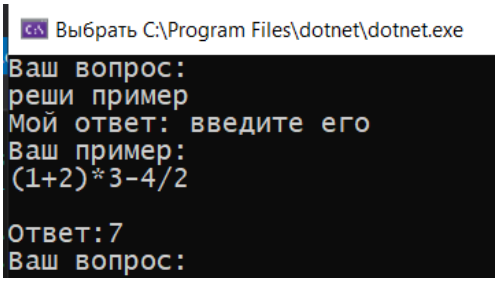

Рис. 9. Тестирование решения арифметических выражений ботом

Таким образом, был реализован простейший чат-бот, который обладает дополнительными тремя функциями, а именно: обучение, проверка на цензуру и решение арифметических примеров [9]. Начинающему программисту легко в нем разобраться, для того, чтобы разработать свою версию чат-бота для новых функций. Дальнейшее улучшение этого бота безгранично, ведь такая задумка позволяет добавить множество разных функций, с помощью которых человеку будет удобнее узнать информацию у бота, чем на других ресурсах [10]. Например, информацию о погоде, перевод текста, построение маршрута и т.д.

Литература

1. Москат, Н.А., Осипова Н.Р. Место систем искусственного интеллекта в современной транспортной отрасли // Сборник научных трудов «Транспорт: наука, образование, производство». Рост. гос. ун-т путей сообщения. Ростов н/Д, 2021. С. 150-153.

2. Лященко З.В, Игнатьева О.В. Современный потенциал развития смарт-технологий и интеллектуальных систем // Сборник научных трудов «Транспорт: наука, образование, производство». Рост. гос. ун-т путей сообщения. Ростов н/Д, 2020. С. 88-91.

3. Zeroual I., Lakhouaja A. Data science in light of natural language processing // Procedia Computer Science. 2018. №127. pp. 82-91

4. Маслова М. А., Бажутова Д. А., Дмитриев А. С. Алгоритмы работы чат-бота для поиска товаров // Инженерный вестник Дона, 2021, № 4. URL: ivdon.ru/ru/magazine/archive/n4y2021/6921.

5. Федутинов К.А. Машинное обучение в задачах поддержки принятия решений при управлении охраной природы // Инженерный вестник Дона, 2021, № 9. URL: ivdon.ru/ru/magazine/archive/n9y2021/7186

6. Куликова Я.В., Качалов Д.Л. Метод определения эмоционального состояния человека при помощи чат-бота // Инженерный вестник Дона, 2022, № 9. URL: ivdon.ru/ru/magazine/archive/%20n9y2022/7893

7. Hocking Joseph. Multiplatporm Game Development in C#. Manning. Питер. 2019. С. 21.

8. Ведерникова О.Г, Москат Н.А. Расширение и использование редактора визуального программирования для разработки виртуальных тренажеров // Инженерный вестник Дона, 2021, № 1. URL: ivdon.ru/ru/magazine/archive/n1y2021/6783

9. Ведерникова О. Г., Гридякина В. И Актуальность применения чат-ботов для сбора данных о пользователях // Сборник научных трудов «Транспорт: наука, образование, производство», Том №1. Технические науки. Рост. гос. ун-т путей сообщения. Ростов н/Д, 2022. С. 43-47.

10. Лященко, З. В., Хусаинов В. Р. Технологии развития искусственного интеллекта // Сборник научных трудов «Цифровые инфокоммуникационные технологии». Рост. гос. ун-т путей сообщения. Ростов н/Д, 2021. С. 68-71.

References

1. Moskat, N.A., Osipova N.R. Sbornik nauchnykh trudov «Transport: nauka, obrazovanie, proizvodstvo». Rost. gos.un-t putey soobshcheniya. Rostov n/D, 2021. pp. 150-153.

2. Lyashchenko Z.V, Ignatieva O.V. Sbornik nauchnykh trudov «Transport: nauka, obrazovanie, proizvodstvo». Rost. gos. un-t putey soobshcheniya. Rostov n/D, 2020. pp. 88-91.

3. Zeroual I., Lakhouaja A. Procedia Computer Science. 2018. №127. pp. 82-91

4. Maslova M. A., Bazhutova D. A., Dmitriev A. S. Inzhenernyj vestnik Dona, 2021, № 4. URL: ivdon.ru/ru/magazine/archive/n4y2021/6921.

5. Fedutinov K.A. Inzhenernyj vestnik Dona, 2021, № 9. URL: ivdon.ru/ru/magazine/archive/n9y2021/7186

6. Kulikova Ya.V., Kachalov D.L. Inzenernyj vestnik Dona, 2022, № 9. URL: ivdon.ru/ru/magazine/archive/%20n9y2022/7893

7. Hocking Joseph. Multiplatporm Game Development in C#. Manning. Питер. 2019. С. 21.

8. Vedernikova O.G, Moskat N.A. Inzhenernyj vestnik Dona, 2021, № 1. URL: ivdon.ru/ru/magazine/archive/n1y2021/6783

9. Vedernikova O. G., Gridyakina V. I Sbornik nauchnykh trudov «Transport: nauka, obrazovanie, proizvodstvo», Tom №1. Tekhnicheskie nauki. Rost.gos.un-t putey soobshcheniya. Rostov n/D, 2022. pp. 43-47.

10. Lyashchenko, Z. V., Khusainov V. R. Sbornik nauchnykh trudov «Tsifrovye infokommunikatsionnye tekhnologii». Rost. gos. un-t putey soobshcheniya. Rostov n/D, 2021. pp. 68-71.