Design, Implementation and Usage of Modern Voice Assistants

Mikhail Belenko, Uliana Muratova, Pavel Balakshin, Nikita Burym
ITMO University
Saint Petersburg, Russia
(mikael0bmv, ulyafkamuratova1, pvbalakshin)@gmail.com, CHVRCHES@mail.ru

Abstract—**As part of this work, some dialogue systems and voice assistants have been studied. Based on their comparison, the basic requirements for creating chat bots were formulated. The well-known open source automatic speech recognition systems were studied and on the basis of comparison the most suitable for integration into the prototype of the chat-bot were selected. The results of the testing of the chat-bot prototype were presented. It is a stand-alone mobile application that works without an Internet connection and allows a dialogue on the day-today topics. The differences between regular and corporate chat-bots were highlighted, and the requirements to corporate chat-bots were collected. Other possible implementation options were briefly considered, for example, for people with speech-impaired voice.**

I. Introduction

The use of natural language processing has been one of the best technological choices to design systems enabling comfortable and useful communication between human and different modern IT devices [1]. This can reduce or even remove the barrier and prejudice that computers are not friendly and it is very difficult to control them in our everyday life. Many IT-companies are developing smart systems to manage home utility systems by voice or simple UI [2]. Also, they compete regularly to create the best programs that people can use to respond to or post information on social networks. For example, in November 2018, IBM hosted a chat-bot for business competitions at ITMO University.

Nowadays many companies have their own websites, messengers, accounts in social networks or even small call-centers. Big IT giants like Facebook, Slack, Discord, Telegram, Kik and Microsoft provide and implement their own chat-bots [3], but the cost of such solutions are pretty high, and they keep all data in their own storages. Such security particularities can contradict with security requirements of many business areas. So, in order not to hire full-time programmers some companies prefer to use outsource programmers and form somehow specifications to build the necessary internal chat-bots. Most of such specifications are private and not published, but initial review of the one such file [4] represents that there is not too much technical information, however, a lot of attention is drawn to the interface. Moreover, it is almost nothing about the main task of the chat-bot: how it will communicate with clients and how to reflect real requirements with a real cases and implementation. Thus, the main purpose of this article is analyze how modern voice assistants can be used and what other modern IT technologies and services they can interact with.

II. Design

A. Terminology and goals

Today there are two known ways to talk to robots: interactive systems (also known as chat-bots) and voice assistants. The chat system or chat-bot can be presented as built-in program or a separate program and usually is designed to use speech or textual methods for dialogue. Therefore, the task of the chat-bot is to identify the person's request, form the necessary commands, and respond to the person. The second technique is called voice assistant. This is a program that can sense and execute voice commands. Such programs are most commonly used in ”smart home” systems and smartphones (tablets).

Due to the similarity of functions and the lack of proprietary names for programs that combine chat-bots and voice helper functions, the correct naming of each type of program is confusing. Programs such as Yandex.Alice are often called chat-bots because they can respond to user requests. Developers refer to such programs as voice assistants, and although these programs combine the functionality of a voice assistant with the functionality of a chat-bot, no name is currently available for such programs. Developers refer to such programs as voice assistants, and as soon as those programs combine the functionality of a voice assistant with the functionality of a chat-bot, no proper and commonly used name is currently available for such programs. However, some researchers call them conversational agents [5].

Voice assistants and voice helpers are two more concepts that are used in the naming of these programs. Their ideas are equivalent, but English word assistant has two translations with different meaning in Russian: assistant and helper. Euler circles in Fig. 1 represents the general functions of these technologies [6].

Based on the above, the research objectives were determined:

1) To study the types and architecture of dialogue systems.
2) To form the basic requirements for creating chat-bots.
3) To test the well-known automatic speech recognition systems (ASR).
4) To compare different neural networks.
5) To develop a prototype chat-bot.
6) To compare the resulting chat-bot prototype with existing solutions.
7) To validate the stated requirements.

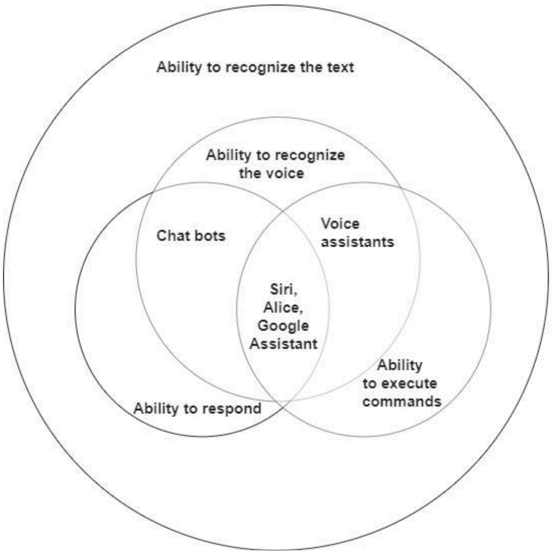

Ability to recognize the text
Ability to recognize the voice
Chat bots
Voice assistants
Siri, Alice, Google Assistant
Ability to respond
Ability to execute commands

Fig. 1. Intersection of communication abilities, provided by common voice applications

B. Comparison of chat-bots and voice assistants

Social network VKontakte, messenger Telegram's chat-bots and Yandex.Alice [7], Google Assistant [8], Siri were chosen for comparison. Comparison was performed by reviewing different scientific articles [9] and official specifications together with testing on mobile phones. Results of comparison and criteria presented in Table I.

Communication is a main chat-bot's function. Chat-bot can communicate only with prepared phrases, so two blocks of phrases should be prepared for any program: human's phrases which chat-bot will be understood and chat-bot's phrases for answers. Worth considering that one phrase can be formulated in various ways (for example, "What time is it?", "What is the hour?" and "What o'clock is it?"), and the answer should be the same. Another researchers figured out that phrase "Hello users!" can be said in 53 ways [9]. However, usage of neural network reduces the set of phrases making problem, which improves the ability of the chat-bot to communicate but complicates the developer's work.

TABLE I. CONSOLIDATED SPECIFICATIONS TABLE OF CHAT-BOTS AND DIALOG SYSTEMS

Criteria	VK's chat-bots	Telegram's chat-bots	Yandex. Alice	Google Assistant	Siri
Count of phrases which the program can understand	less than 10	less than 200	not limited	not limited	not limited
Count of answers	less than 10	less than 50	not limited	not limited	not limited
Using of ASR	-	-	+	+	+
Simple actions	less than 5	less than 20	more than 100	more than 100	more than 100
Integration with other programs	-	-	+	+	+
Independent application	-	-	+	+	-
Other developers can add their features	-	-	+	+	+
Speech synthesis module	-	-	+	+	+

Nowadays people are faced with the lack of time, which in relation to chat-bots is manifested in their unwillingness to type and read text. In such a case the use of a chat-bot can help to make it more usable, which allows a person to perform several actions at once: do personal business and simultaneously communicate with the chat-bot.

Many chat-bots and voice assistants currently cannot work without ancillary programs. So, chat-bots worked in different social networks are designed exclusively for these social networks and do not work outside of them. Even the well-known Yandex.Alice, which is considered a third-party application, is supplied exclusively with Yandex.Browser. For many users this is unnecessary functionality that complicates the work with the application, and they refuse to use chat-bots and voice assistants. In this regard, technology adaptation for a variety of purposes [10] or development of a separate application becomes more relevant.

Multi-criteria comparison was conducted, that included end-user requirements, technological constraints, gathering surveys from 40 master students of ITMO University together with wishes of two IT directors (telecom and construction companies). Based on it, general basic requirements were formulated to which any chat-bot should meet. However, some specific topical areas can either increase or slightly decrease values below.

Minimum requirements:

- Up to 30 phrases with simple syntax that a chat-bot can understand.
- Up to 15 phrases with simple syntax that a chat-bot can answer.
- Implementation in any programming language for any social network.

Sufficient requirements:

- More than 200 phrases that a chat-bot can understand.
- More than 50 phrases that a chat-bot can answer.
- Perform simple operations (for example, record / output of the required data).

Extended requirements:

- Integration with any ASR system.
- Independent application.
- Using neural networks to understand the request and respond to it.
- The ability to add operations.

C. Comparison of ASR systems

There are several well-known commercial ASR. These are Google Cloud Speech, Amazon Alexa, IBM Watson, Siri, and Yandex.SpeechKit, Microsoft Speech to Text, etc. However, these systems have drawbacks that prevent them from being used to develop separate personal application:

- all these ASR are working with the Internet only, so applications can be less useful because without the Internet the ability to speech recognition will be unavailable or will be work with long delay;
- all these ASR are oriented on normal use, and they can't work with the terms in specific areas;
- with it can be problems with privacy and information protection – anyone who has access (including illegal) to the server can listen to the recording sent for recognition.

Based on the foregoing, open source speech recognition systems are chosen because this allows configuring and training them as on the needs of the developer and user.

The most used and well-known open source ASRs are Kaldi, Mozilla DeepSpeech [11], Sphinx, RWTH ASR, and Julius [12]. The criteria and comparison of the listed ASR are presented in Table II.

Based on the known and previous research [13] Julius system has the highest percentage of errors (Word Error Rate – WER metric), while the DeepSpeech and RWTH ASR systems have the longest time for recognizing of speech recording (Speed Factor metric). Thus, it is better to draw more attention on Kaldi and CMU Sphinx. The first is convenient for use in a client-server application and allows to calculate and compare different metrics, which is necessary to understand the quality of ASR work and a chat-bot in general and, if necessary, improve this work. The second allows to create an application for mobile phone without usage of additional funds and work with the chat-bot offline. However, only CMU Sphinx is designed for use in applications designed for the Android operating system, and since the chat bot is most convenient to use with a smartphone, it is with this operating system should work ASR. Based on the above, for the development of a prototype chat-bot will be selected CMU Sphinx speech recognition system.

TABLE II. SUMMARIZED PERFORMANCE TABLE OF SELECTED ASRS

Criteria	**Kaldi**	**DeepSpeech**	**Sphinx**	**RWTH ASR**	**Julius**
WER, %	6,5	7,2	21,4	15,5	23,1
SF	0,6	4,3	0,5	3,8	1,3
Programming language	C++	Python	C/Java	C++	C
Structure	Modular	Modular	Modular	Modular	Modular
Supported OS	Linux, Windows, FreeBSD	Linux, Mac OS, Windows, Android	Linux, Mac OS, Windows, Android	Linux, Mac OS	Linux, Windows, FreeBSD, Mac OS
Interface	Cantilevered	Cantilevered	Cantilevered, API	Cantilevered	Cantilevered, API
Languages	English	English	Many languages, including exotic ones	English	Japanese, English
Speech Recognition Research	Intended	Need to write modules if necessary	Need to write modules if necessary	Need to write modules if necessary	Need to write modules if necessary

III. Implementation

A. Selection of development tools

Before the start of a chat-bot developing, there is a necessity to understand which platform a program will be written and what tools will be used in the process developments.

The most convenient and frequently used chat-bots and voice assistants, that use ASR are written for smartphones. Accordingly, it is necessary to use development environment that takes into account all features of the Android operating environment. Of the two currently known development environments (Android Studio and Eclipse) more modern, convenient and used is Android Studio. This development environment has all the necessary tools, including the smartphone's emulator to run and verify that the application is working. So Android Studio allows to be safe by running your own program's intermediate results on emulated devices, that also takes less time than installing the same application on a real phone.

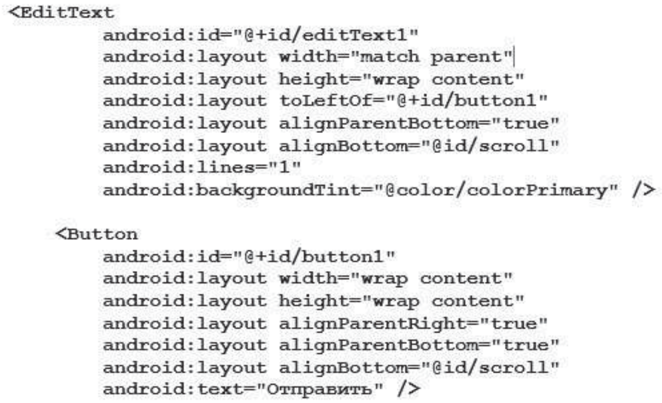
It is also necessary to determine the programming language. Several languages are used in pair with Android OS, but at moment the most common and popular are Java and Kotlin languages [14]. Java has a lot of documentation, various articles and reference books, forums, special community, so this programming language is preferred for use there.

B. Chat-bot prototype development

The following stages of bot creation can be distinguished:

- Creating a graphical part of the bot.
- Creating a functional part of the bot (ability to respond to human phrases).
- Addition of ASR.

Graphics implementation is the first thing to do, i.e. how the bot will look like. In order for a person to be able not only to speak but also to write, you need to add a field for entering text and a button to send the entered text. The EditText and Button components are used for this purpose. For ease of use, their location at the bottom of the screen is indicated. The file activity-main.xml is used to indicate the markup. Some lines are presented in Fig. 2.



```
<EditText
        android:id="@+id/editText1"
        android:layout width="match parent"
        android:layout height="wrap content"
        android:layout toLeftOf="@+id/button1"
        android:layout alignParentBottom="true"
        android:layout alignBottom="@id/scroll"
        android:lines="1"
        android:backgroundTint="@color/colorPrimary" />

    <Button
        android:id="@+id/button1"
        android:layout width="wrap content"
        android:layout height="wrap content"
        android:layout alignParentRight="true"
        android:layout alignParentBottom="true"
        android:layout alignBottom="@id/scroll"
        android:text="Отправить" />
```

Fig. 2. Example of configuration in XML file

The next step is to train the bot to respond. First it's necessary compile a sample list of phrases that will be used when working with a bot. Everyday language phrases were chosen in order to implement working prototype that can be easily tested. The most convenient way for the bot to work is to use the xml markup file. Now it is possible to start creating a dialog. The steps that are used for chat-bot's work can be formulated as algorithm 1.

Algorithm 1 Chat-bot's execution algorithm

```
0: A person enters a phrase, clicks on the "Send" button.
0: The bot reads the person's phrase, displays it on the screen.
0: The bot searches for a person's phrase in the phrases.xml
   file and reads the answer phrase.
0: The bot prints the answer phrase on the screen.
0: If there is no human passphrase in the file, print out prepared
   response
```

Methods addPhrase() and getAnswer() are mostly used to code chat-bot's work. These methods are written to make the bot work properly, so it is necessary to describe them in more detail.

The addPhrase() method is used to display any phrase on the screen. To display a phrase on the screen, we need to know the background and text colors, the phrase and so on, whether it's a bot or a person. The new TextView (element allowing simply to display the text on the screen) is created to output each new phrase. Creation of new TextView element for each new phrase is necessary for to save a dialog with the chat-bot on the screen. Then parameters are set for this element (indents, text and background colors, text alignment depending on the speaker), and it is displayed on the smartphone screen.

The getAnswer() method is used to read the phrases.xml file and select of the bot's response. The first thing to do is to download the file from which the bot will be to read the answers. Then there's a tagging breakdown. First it looks for the tag human_phrase, saying that the phrase was made by a man. Then the text contained in the tag is read. If the text is the same as the phrase that has been set by a human being and has been read out before, then it is necessary to find the first bot_answer tagged phrase following the human phrase found, and return it as a result of the method. If the current human found the phrase does not coincide with the specified one, it is necessary to continue searching through the file.

Now it is possible to begin to connect to the system project speech recognition. Since the chat-bot uses the Russian language, the data needed to of the Russian language. From the project website (https://github.com/cmusphinx/pocketsphinxandroid-demo) one needs to download the recognition module itself in addition to a linguistic model and dictionary for the Russian language. Using developer's manual (https://cmusphinx.github.io/wiki/tutorialandroid/), it is possible to connect the module to the current project. Then, based on the file phrases.xml it is necessary to write a grammar for the bot [15]. Received grammar can be seen in the application.

Now it is necessary to add the required methods for the chat-bot to work. First one needs to check if the application has access to a microphone and audio recording, because without it the application will not be able to work. Then you can start working with the resolver. At this stage it is required to describe the recognition parameters (including the location of the language model files – grammar, dictionaries, etc.) and allow ASR to work in the background, so that the chat-bot can work at any time without pressing an additional key or button. In case of closing the application, the user does not need to listen to it or consume any unexplained resources, so it is necessary to turn off the resolver.

If the phrase is recognized successfully, a bot reaction is required, so after the result of the recognition appears, the bot searches for the answer to the resulting phrase using known methods and displays the answer on the smartphone screen. Also, all the necessary files, directories and activation phrases are installed for the correct initialization of the recognizer. At the end it is necessary to describe the reaction of the program in case of an error. In case of correct configuration, the result can be presented on mobile's screen (Fig. 3). Presented discussion was translated into English for better vision and understanding of implemented chat-bots results (Fig. 4).

Released chat-bot has 229 phrases which ASR can understand, 60 answers and processing as an independent stand alone mobile application. Those numbers were defined to exceed sufficient requirements described above: 200 phrases to understand and 50 phrases to answer.

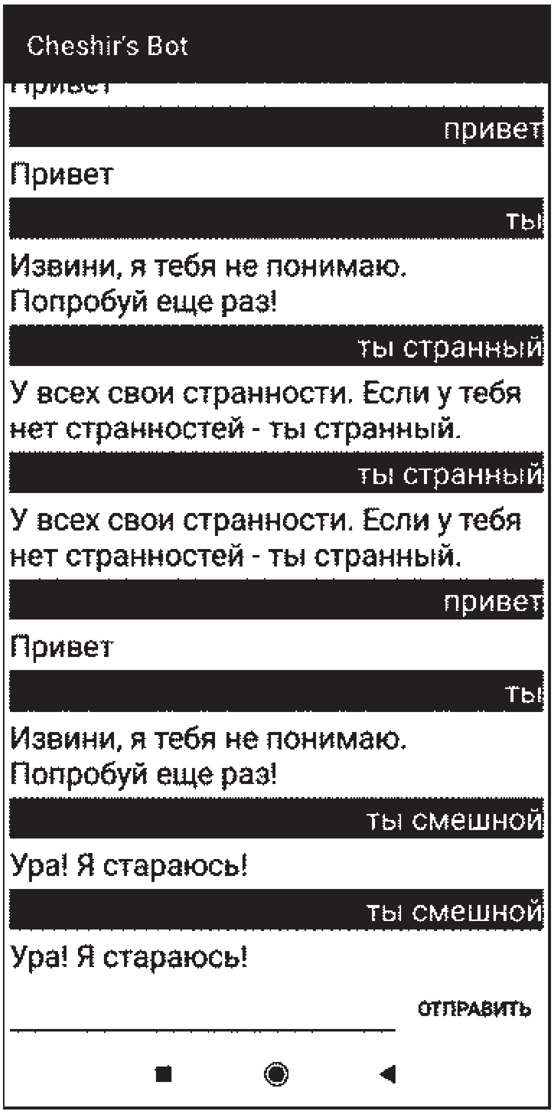
C. *Testing and future comparison with analogues*

Manual chat-bot testing was carried out for validation and verification by three different smartphones' usage: Xiaomi Redmi Note 6 Pro, Xiaomi Redmi 4, Huawei P10 Lite. Testing was conducted by four people: two men of 21 and 49 years old and two women of 21 and 49 years old as well. Russian is the first language for all of them. Of course, it is much better to have at least 10 testers, but the main idea was to prove stable and consistent work of stand-alone chat-bot mobile application combined from multiple third party components. In the course of the testing, it was found that all designed requirements were fully met. There are plans to collect and analyze additional user evaluation like self-report questionnaires, user observations, logs, etc. Also, in the process of testing it was found out that joint work of compliance-oriented modules allows the application to achieve the main purpose of its creation – to be able to have quite real communication with humans.

The development resulted in a prototype of a chat-bot capable of communicating with humans. To estimate the level of the prototype's work it is necessary to make a comparison with previously selected chat-bots and voice assistants. One of the tasks of this work was to develop a program that would be able to communicate with the person in a text form and recognize the voice of the user, so part of the functionality (such as the ability to execute commands) was deliberately ignored.

IV. Possible implementations

One of the possible usage of speech technologies is a corporate chat-bot creation. Even a quite small companies are considering partially replace some internal operators with computer devices capable of recognize speech, then process



Привет

привет

Привет

ты

Извини, я тебя не понимаю.
Попробуй еще раз!

ты странный

У всех свои странности. Если у тебя нет странностей - ты странный.

ты странный

У всех свои странности. Если у тебя нет странностей - ты странный.

привет

Привет

ты

Извини, я тебя не понимаю.
Попробуй еще раз!

ты смешной

Ура! Я стараюсь!

ты смешной

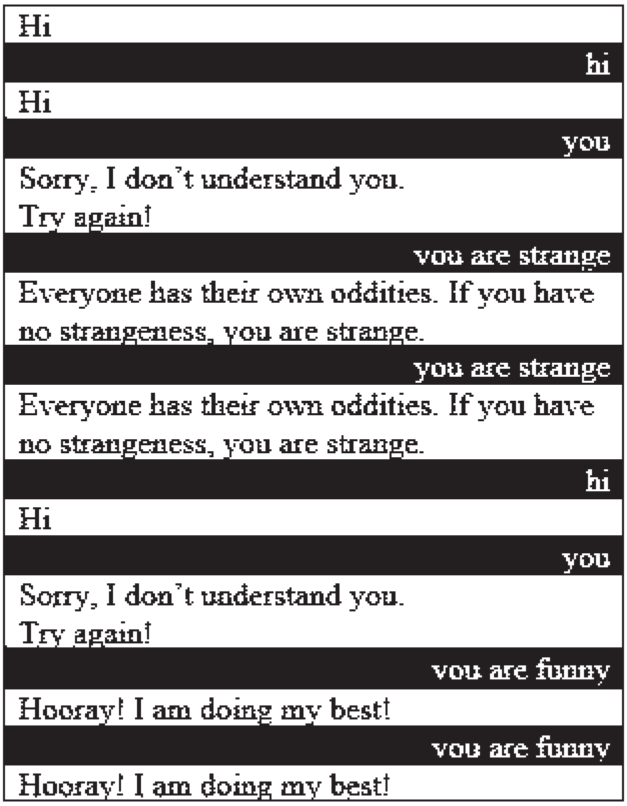
Ура! Я стараюсь!

Fig. 3. Result of chat-bot's work

and analyze it, and as a result provide a meaningful and necessary response to the client [16]. Proper usage of such software will allow to increase the efficiency of data processing and company operating. Thus, it is very important to fully and thoroughly understand peculiarity of corporate chat-bots prior to further investigations.

The corporate chat-bot is intended for using for business companies which is reason for some additional conditions:

- Firstly, it requires high security level to guard confidential information, at a minimum, the use of public resources must be excluded. Even usage of well-known cloud services like Amazon is prohibited.
- Secondly, it needs to develop a application which can support a sufficiently large number of simultaneous requests. It's necessary for a big companies with much people, because in one time the most people need to use the chat-bot. In this situation the application must handle the load without failings.



Hi

hi

Hi

you

Sorry, I don't understand you.
Try again!

you are strange

Everyone has their own oddities. If you have no strangeness, you are strange.

you are strange

Everyone has their own oddities. If you have no strangeness, you are strange.

hi

Hi

you

Sorry, I don't understand you.
Try again!

you are funny

Hooray! I am doing my best!

you are funny

Hooray! I am doing my best!

Fig. 4. Translated example of discussion with released chat-bot

- Thirdly, the chat-bot must communicate on abstract topics (it can be interesting for relaxing of employees) and must answer on basic questions from employees of this company. In this time ability of doing some routine without people will be more comfortable for employees, for example, generating a report by a template or adding an appointment to the calendar. In this case, additional research is required on the products that companies use to integrate chat bots with those products. It can be ability to add code for open source programs or using API for proprietary programs.

Also, there are multiple restrictions and special requirements that will led corporate assistance to be a special software including features of chat-bot and voice assistant. Chat-bot functions are necessary in any case, since the developed program must accept requests and respond. Integration of voice functions can be useful in out of office or when user don't have time for texting. For example, during an accident or during urgent preparation of multiple documents. This function is more useful on mobile devices like smartphones and laptops. Sometimes it may be useful on notebooks. However, using of chat-bots will be minimal in offices like open space when voice control can distract any of the colleagues nearby.

It is necessary to think over the architecture and the used parts of the chat-bot in advance, depending on the necessary capabilities:

- At first, dictionaries should be developed when creating a chat-bot. Those dictionaries will be used for neural network training or for use in chat-bot itself directly. They must contain usual phrases and terms which uses in company. This step should increase speech recognition and the usefulness of the application in general.

- At second, it is necessary to preserve the possibility of vocabulary extension and the possibility of neural network re-training based on usage history. This can be done in two different ways – save the usage history and periodically send the saved data to the developers, or organize full work of the application on the Internet.
- At third, it needs to choose ASR and language for using in chat-bot. It's comfortable when the ASR will be on server because loading the program with ASR on the local device takes a long time. It can be critical for user opinion about application. So it can be any ASR in client server application, for example, Kaldi or Sphinx. But Kaldi hasn't a Russian language support. It can be difficult for using chat-bot in Russian companies.

Another important area for chat-bot usage is creation of specific application for people with speech impairments. Such application can prove and clarify speech disorders, precise impaired speech classification and can further improve the quality of communication between human and different IT devices.

V. Conclusion

The development of technologies related to artificial intelligence today is very rapid, but a person must do much more work to achieve this goal, including the field of speech recognition and computer training.

The one of the studies from this area is this work, which examines and compares several chat bots and voice assistants. Based on this comparison, the general requirements for chat-bots were formulated. Accordingly, those requirements can be included in the specifications for the creation of chat-bots, and similar requirements will raise the level of developed chat-bots and allow to get more profit from theirs work.

Also, the most well-known open source speech recognition systems were considered, their comparison was carried out and the optimal one was selected for use in the ASR mobile application.

Based on the characteristics comparison and results of the released bot testing, it can be argued that the developed prototype is at the level of Telegram and VKontakte chat-bots, and in some characteristics even superior: the prototype does not use the Internet and other additional resources, has a larger set of phrases than most social networking bots. However, compared to more advanced voice assistants, the prototype is currently losing most of its characteristics: the limited number of phrases and the lack of a built-in neural network do not allow the bot to be trained and make it less predictable. But overall, all four testers were satisfied with the results.

In the future, it is planned to continue and expand this work, including research on the use of neural networks and the development of other chat-bots: for people with impaired speech and effective corporate one. That will require involving additional test users, creating detailed test plans and various aspects of personal evaluation.

Acknowledgment

This work done with support of ITMO University project # 619296 "Development of methods for the creation and implementation of cyber-physical systems". The authors would like to thank Ph.D. associate professor of the Software Engineering and Computer Technology department N.M. Lukyanov for his help in corporate chat-bots research.

References

[1] *M*. Zajechowski, Automatic Speech Recognition (ASR) Software – An Introduction, Web: https://usabilitygeek.com/automatic-speech-recognition-asr-software-an-introduction/.

[2] *V*.N. Shmatkov, V.N. Bakowski, D.S. Medvedev, S.V. Korzukhin, D,V. Golendukhin, S.F. Spynu, D.I. Mouromtsev, "Interaction with Internet of Things devices by voice control", *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, vol.19, no.4, 2019, pp. 714–721 (in Russian). doi: 10.17586/2226-1494-2019-19-4-714-721.

[3] *N*.A. Tugusheva "Chat-bot use in various spheres of daily life", *Young scientist*, vol.21 (155), 2017 (in Russian).

[4] "TERMS OF REFERENCE for the development and implementation of chat-bots in instant messengers: "Viber", "Telegram", "Facebook" and in the social network "VKontakte" for, "Facebook" and in the social network "VKontakte" for "and in social VKontakte network for consumers of MSW management services", *Khanty-Mansiysk*, 2019.

[5] *X*. Yang, M. Aurisicchio, W. Baxter, "Understanding affective experiences with conversational agents", *in Proc. 2019 CHI Conference on Human Factors in Computing Systems, CHI 2019*, May 04-09, 2019. doi: 10.1145/3290605.33007721

[6] *M*. Belenko, N. Burym, U. Muratova, P. Balakshin, "Training aspects of automatic speech recognition systems during chat-bot creation", *19th International Multidisciplinary Scientific GeoConference, SGEM 2019*, vol.19, is.2.1, pp.681-688.

[7] *A*lice's Skills, Web: https://dialogs.yandex.ru/store.

[8] *G*oogle Assistant, Web: https://assistant.google.com/explore?hl=ru ru.

[9] *A*.V. Paraskev, A.A. Kadentseva, S.I. Moroz, "Prospects and peculiarities of the chat-bots development", *Scientific journal of Kuban State Agrarian University*, vol.130 (06), 2017 (in Russian).

[10] *A*. Bhalla, "An exploratory study understanding the appropriated use of voice-based Search and Assistants", *in Proc. 9th Indian Conference on Human Computer Interaction*, 2018, pp. 90-94.

[11] *M*.V. Belenko, P.V. Balakshin, "Comparative analysis of speech recognition systems with open code", *International Research journal*, vol.4 (58), 2017.

[12] *D*. Boyko, Open source speech recognition systems, Web: https://lvee.org/en/abstracts/273.

[13] *U*.D. Muratova, P.V. Balakshin "Development of the requirements for the chat-bots creation", *Proceedings of VIII Congress of young scientists, 2019*, vol.3, pp. 285-289.

[14] *A*.V. Prendota, P.V. Balakshin "Component in the Kotlin programming language for integrating exe-cutable programs into internet resources", *Software & Systems.*, vol. 32, no. 4, 2019, pp. 690–695 (in Russian). doi: 10.15827/0236-235X.128.690-695.

[15] *S*peech recognition with the help of CMU Sphinx (in Russian), Web: https://habr.com/ru/post/267539/.

[16] *D*. Khizhinsky D. Speech recognition will blow up the call-center market (in Russian), Web: http://www.cnews.ru//index.shtml?2007/05/22/251191.