Russian Language Neural Net Chatbot with Natural Language Processing

Nurullo Ismoilov
Division for Experimental Physics
Tomsk Polytechnic University
Tomsk, Russian Federation
nii4@tpu.ru

Mikhail Semenov
Division for Experimental Physics
Tomsk Polytechnic University
Tomsk, Russian Federation
sme@tpu.ru

***Abstract* – In this paper, we consider a chatbot, which can reply to various user commands and uses natural language processing. Moreover, the most common employee's working processes were automated. This solution can work under any corporate local or global networks. Although, in this article, used tools, software and libraries are explained as well. As a result, chatbot prototype is presented.**

Keywords –chatbot, machine learning, natural language processing

I. INTRODUCTION

Today the large companies look for ways of introduction of intellectual assistants that are capable to carry out various routine office tasks. These are some examples of theese tasks: staff recruitment, training of new employees, answer to questions, paperwork, planning of time and resources [16]. Intellectual assistants based on machine learning (ML) allow computer programs to learn from data.

A computer program is said to learn from experience E with respect to some task *T* and some performance measure *P*, if its performance on *T*, as measured by *P*, improves with experience *E* [4].

Data from which computers learn is called *training set* and each training example is a training sample.

Machine learning is great for next problems:

- Problems for which existing solutions require a lot of code to set up rules, where a ML algorithm can either perform better and simplify the code.
- Complex problems for which there is no good solution at all by using a traditional programming approach, the ML could be a good solution.
- Fluctuating environment. A ML system can adapt a new data
- Getting insights about complex problems and large amounts of data.

It is a fact that companies around the globe are using ML in their business, because it allows them with less investment gain more income and spend less time for the same task.

Chatbots are one of the representatives of modern business solutions. They are getting more popular these days, because they are able to work for 24 hours 7 days a week.

Report provided by Global Digital for 2019 [5] show that 4.021 billion people are using messengers while 3.196 billion are using social networks and the number of messengers' users are increasing by 79 % more than social networks' users. The active distribution of instant messengers as a tool for easy and quick messaging began in 2015.

People today prefer using messengers, as they are more efficient and less time consuming for business processes and everyday tasks.

Most messenger's API's are providing a chatbot solution. The first chatbot was introduced to the world in the 1960s by MIT professor Joseph Weizenbaum. It was called ELIZA. It could breed a psychologists' speech by finding out keywords or either say "okay" and transfer the conversation to other direction if it didn't know the answer.

Chatbots today are very different from ELIZA. They can solve more tasks, use natural language processing (NLP) to understand commands and use ML to improve themselves. There are four types of chatbots:

1. **Click navigation chatbots.** These kind of chatbots work with preloaded «if — else» logic, created through a pre-designed conversation.
2. **Keyword based chatbots.** These chatbots use a huge list of keywords database. They answer to the user question if the keyword is in question.
3. **Natural language understanding chatbots.** These types of chatbots use neural networks to understand natural language and to form answer to the users or do their tasks.
4. **Aware chatbots.** This kind of chatbots are not developed yet, since for now it is impossible. It is bots like JARVIS or FRIDAY from Marvel movies.

Existing chatbots have one common deficiency: they cannot be installed on a company's server, since they use Messenger's API, so that they work only with that messenger.

In this paper we build a messenger based chatbot, which will use Machine learning, natural language processing and Markov chain.

More than 50 % of the internet content is in English language while Russian segment is less than 10%. Giants like Google, Microsoft and IBM are presenting developer's tools mostly in English. That is why the accuracy of chatbots in English are higher than in any other language.

The other problem is Acronyms. Acronym is a word or name formed as an abbreviation from the initial components of a phrase or a word, usually individual letters. NLTK library already contains list of acronyms in English language, which

978-1-7281-2896-2/19/$31.00©2019 IEEE

means that if you type NATO computer will understand that it is North Atlantic Treaty Organization. In case of Russian language, it won't understand such words and it will cause the accuracy to get lower.

The solution is to create own acronyms list and add to the NLTK library.

We have created a prototype of a «Natural language understanding» Russian language chatbot. Our final aim is a chatbot that can be integrated with corporate networks regardless of the operating system and will improve productivity. For achieving our aim, we have such tasks:

a) analyze the existing tools for developing our chatbot,

b) develop a console chatbot prototype,

c) test chatbot by chatting and giving tasks,

d) develop a user interface for chatbot,

e) integrate chatbot with a local network.

II. CORPORATE TASKS ANALYSIS AND DEVELOPMENT TOOLS

Corporate employees are facing a lot of tasks to solve. For making such regular task like making a travel order an employee should write about it to a business travels' office, they should find tickets and hotels and coordinate with top-brass, after they reserve the ticket and the hotel and send confirmation to the employee. All of this process could be automated by a bot, who could do this task a lot faster. Other example is, if some references or inquiries are needed a chatbot could send it to printer and send message to the working stuff that a stamp is needed. All of this will save a lot time and finally it brings good profit to organizations.

To create a chatbot existing tools were examined. There are lots of tools created to build chatbots [6-10]. Morph.ai [6] is a platform that allows to create a business chatbot which uses WhatsApp, Facebook messenger or a website as a body of the bot. Such bots understand natural language. Data from such bot cannot be store on local network of the organization which can bring to corporate information leak.

Flowxo [7] positioned as a zero coding chatbot, which allows to build chatbots as fast as it is possible. It also can be integrated with Gmail, LinkedIn and JIRA. It has the same flaw as previous chatbot.

Telegram.API [8] allows to build a telegram chatbot. It has lots of pluses, but the deficiency is that it uses telegram as a bot platform.

API.AI [9] allows to build a learning chatbot which can be adopted with Android, iOS, Node.js and HTML. But the main problem is in free version there're lots of limits such as the number of requests or messages. Also, the company cannot be sure about confidential information if it is safe.

Python [10] is a simple yet powerful programming language and it has lots of libraries which allow to create a machine learning and natural language understanding chatbot these tools are free to use. Python allows to connect own database to the user which can be stored on the company's servers.

In this paper, we have chosen Python as a programming language. From the instruments we use following tools:

- NLTK library (natural language processing tool kit) [1]– which was created in 2001 as a part of computational linguistics course in the Department of Computer and Information Science at the University of Pennsylvania. Now it is developed and used as a data analysis tool all around the world.
- pymorphy2 [2] is a morphological text analyzer which was developed in Python and allows to lemmatize and analyze words, divide them by grammatical characteristics. It uses with Open Corpora dictionary and makes hypothesis for unknown words. Russian and Ukrainian languages are supported.
- TensorFlow [11] is an end-to-end open source platform for machine learning developed by Google. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers to work with ML.
- Keras [12] is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK or Theano. It was developed with a focus on enabling fast experimentation.
- NumPy [13, 14] is the fundamental package for scientific computing with Python. It contains among other things: a powerful *N*-dimensional array object (tensors), broadcasting functions, tools for integrating C/C++ and Fortran code, useful linear algebra, Fourier transform, and random number capabilities.
- Markov Chain [17] is a mathematical system usually defined as a collection of random variables, that transition from one state to another according to certain probabilistic rules. These set of transitions satisfies the Markov Property which states that the probability of transitioning to any particular state is dependent solely on the current state and time elapsed, and not on the sequence of state that preceded it. An example of Markov chain can be seen in figure 1.

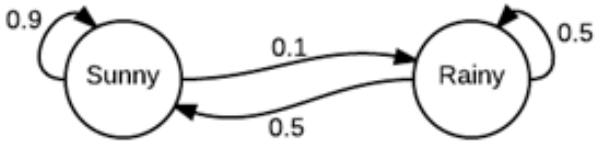

Fig. 1. Markov chain example with two events

- Word2vec [18] is a group of related models that are used to produce word embeddings. These models are shallow, two-layer neural networks that are trained to reconstruct linguistic contexts of words.

For the chatbot we construct a ML neural network. A machine learning system is *trained* rather than explicitly programmed. It is presented with many *examples* relevant to a task, and it finds statistical structure in these examples which eventually allow the system to come up with rules for automating the task [3].

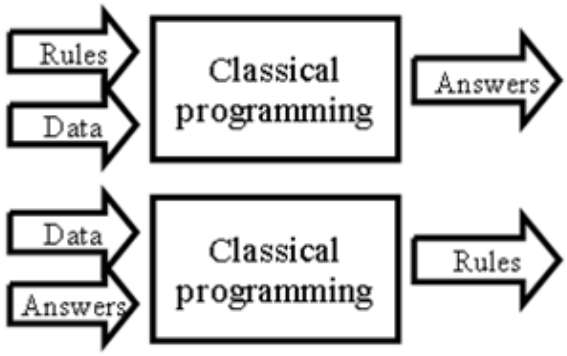

Fig 2. Machine learning concept

There are two ways to do it:

1. To build a neural network for each task. In our case these tasks are to interpret the language, to track the set of a conversation and to generate a response. Each of these systems will work separately to do its own task. This all is unnecessarily complex to build and very time consuming to train such systems.
2. The better type of building a neural network is called «End-to-End» Systems. These are chatbots that use one system that trained on one dataset. They make no assumptions about the use case or the structure of a dialog. It should be just trained on a relevant data. After training it will be able to chat with us about this data using natural language.

In this paper we build an «End-to-End» system. All data for training is downloaded from the Russian Language Corpus in w2v format [18]. Word2vec library is used to read that data. The dataset is consisting of 14 Gb words converted and stored as vectors.

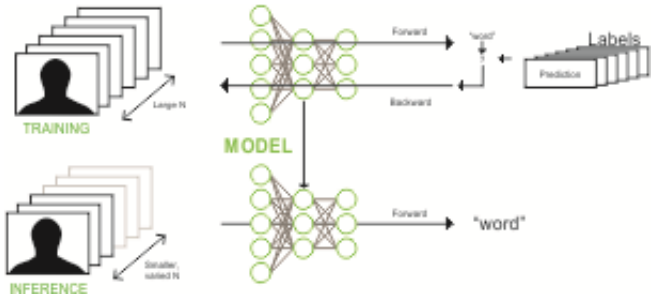

Fig. 3. End-to-End Systems

III. IMPLEMENTATION

In this step our chatbot prototype is working in command line interface. The incoming queries are processed with NLTK library [1]. In the first step we use «Tokenization». It is a process that splits an input sequence into so-called tokens. Token is a useful unit for semantic processing. It can be a word, sentence or paragraph. NLTK library provides a whitespace tokenizer, punctuation tokenizer and TreeBankWord tokenizer. Tests show that whitespace tokenizer is the best for the Russian language. For higher accuracy we first use sentence tokenizer, than whitespace tokenizer to achieve higher accuracy.

The second step is token normalization. In this step *stemming* or *lemmatization* is used. Stemming is a process of removing and replacing suffixes to get to the root form of the word, which is called stem. Lemmatization usually refers to doing things properly with the use of a vocabulary and morphological analysis. It returns the base or dictionary form of the word which is known as lemma. NLTK library allows to do this, but the problem is, the Russian language accuracy is around 15 %, which is very low. For this reason, we have used the pymorphy2 library. The accuracy is around 60 %. It is the best possible solution for now.

Next, we have added our own command list into NLTK database which allows our bot to understand some slang words (for instance, set a birthday do on Friday).

After this we should find main words in the user's query. For this we have to count TF-IDF (term frequency-inverse document frequency) value. It is a numerical statistic that is intended to reflect how important a word in a document in a collection or corpus. We have counted the TF-IDF value as follows [19]:

$$TF = \frac{f_{f,d}}{\sum f_{t\prime d}} \quad (1)$$

$$IDF = log\frac{N}{|\{d \in D: t \in d\}|} \quad (2)$$

$$TFIDF(t,d,D) = TF \cdot IDF \quad (3)$$

where $f_{f,d}$ – frequency for term in document d, $\sum f_{t\prime d}$ –total number of words in the document, D – number of documents in the collection, $|\{d \in D: t \in d\}|$ – number of documents in D, which contains t.

A high weight in TF-IDF is reached by high term frequency in the given document (input data) and a low document frequency of the term in the whole documents (database). The word with the highest weight of TF-IDF value is the main word. In Python to count TF-IDF value sklearn and pandas libraries are used. A code fragment is in listing 1.

```
>from sklearn.feature_extraction.text import TfidfVectorizer
>import pandas as pd
>text=input().split()
>tfidf=TfidfVectorizer(min_df=2, max_df=0.5, ngram_range(1,2))
>features=tfidf.fit.transform(texts)
>pd.DataFrame(features.todense(), columns=tfidf.get_feature_names())
```

Listing 1. Counting TF-IDF values

The problem for the TF-IDF value is lots of data can slow down the work of the chatbot. For example, we have an input from the user and as a result we have a matrix of features consisting of 25 000 rows and 75 000 columns, that's really a huge matrix of data. As we look to that matrix up to 99.8% of TF-IDF values are zeros. We apply machine learning model to apply some restrictions. To solve this we apply logistic regression, which works as following: It tries to predict the probability of the TF-IDF value being in a given diapason, so that searching will be much easier. Depending on TF-IDF values the responses are grouped.

As we know what the user wants from our chatbot, we can reply it. The reverse process of answering begins. Using

Russian language corpus, the answer is formed. To answer questions mostly it uses texts from sentences in the corpus. Also, it is possible to correct chatbot's answer. It also able to learn from corrected data as well.

For now, the chatbot is working from terminal on Ubuntu Linux OS and able to send e-mail using SMTP protocol from user's e-mail address.

Current answering accuracy of the bot is about 55%. Further optimization of the bot can cause the accuracy to go up. It is because Russian language segment in NLTK and other libraries are not developed good enough. The solution can be connecting the bot directly to the Russian language corpus as if it could learn directly from data collected there and also using DeepPavlov open-source conversational AI library.

DeepPavlov is built and maintained by Neural Networks and Deep Learning Lab at MIPT within iPavlov project (part of National Technology Initiative) and in partnership with Sberbank [20]. It has comprehensive and flexible tools that let developers and NLP researchers create production ready conversational skills and complex multi-skill conversational assistants.

IV. Conclusion

In this paper we developed a chatbot prototype using NLTK, NumPy, ChatterBot and Word2vec data, which can analyze user input and reply to it in natural language, send an e-mail with given text. As a basic language were chosen the Russian language.

Future improvements could be first of all by using TensorFlow and Keras libraries and DeepPavlov for Russian language accuracy improvement. Secondly Markov chain will be used to generate answers to the user input. Also, it is planned to add features like adding up an event to the calendar, automatic document creation, printing documents, tickets reservation and sending SMS.

Approaches used in this work could be used for English and other national languages.

References

[1] S. Bird, E. Klein, E.Loper. *Natural language processing with Python.* Sebastopol, CA: O'Reilly Media Inc., 2009, pp. 179-215.

[2] K. Mikhail. "Pymorphy2 documentation". Internet: https://pymorphy2.readthedocs.io/en/latest/user/ [June, 10, 2019]

[3] F.Chollet. *Deep learning with Python.* Shelter Island, NY: Manning publications co., 2018, pp. 25-49.

[4] A. Geron. *Hands on machine learning with Scikit-Learn and TensorFlow*. Sebastopol, CA: O'Reilly Media Inc., 2009, pp. 291-323.

[5] "Global Digital" Internet: www.wearesocial.com [August, 7, 2019]

[6] "Morph.ai."Internet: www.morph.ai [May, 8, 2019]

[7] "Flowxo." Internet: www.flowxo.com [May, 8, 2019]

[8] "Telegram API." Internet: https://tlgrm.ru/docs/bots/api [May, 8, 2019]

[9] "API.AI." Internet: https://dialogflow.com [May, 8, 2019]

[10] P. Barry. *Head first Python.* Sebastopol, CA: O'Reilly Media Inc., 2016, pp.182-230.

[11] "TensorFlow." Internet: https://www.tensorflow.org [February, 22, 2019]

[12] "Keras." Internet: https://keras.io [February, 22, 2019]

[13] W. McKinney. *Python for Data Analysis.* Sebastopol, CA: O'Reilly Media Inc., 2016, pp.32-66

[14] "NumPy." Internet: https://www.numpy.org [February, 22, 2019]

[15] "ChatterBot." Internet: https://chatterbot.readthedocs.io/en/stable/ [February, 22, 2019]

[16] G.Chursin, A. Dorzhiev, N. Ismoilov. "Embedded Chatbot in Corporate Systems." *Prospects of Fundamental Sciences Development*, vol. 3, pp. 92-94, April 2019

[17] C. M. Grinstead, J. L. Snell. *Introduction to probability*. Rhode Island: American Mathematical Society, 2015, pp.405-470.

[18] "Word2vec. Corpus of Russian Language." Internet: https://nlpub.ru/Russian_Distributional_Thesaurus

[19] "Natural Language processing." Internet: https://www.coursera.org/learn/language-processing/lecture/T7fNB/linear-models-for-sentiment-analysis [January, 26, 2019]

[20] "DeepPavlov." Internet: https://deeppavlov.ai/ [February, 26, 2019]