Natural language processing implementation on Romanian ChatBot

RALF FABIAN, MARCU ALEXANDRU-NICOLAE
Department for Informatics
“Lucian Blaga” University of Sibiu
Ion Rațiu Street, no. 5-7, Sibiu
ROMANIA
ralf.fabian@ulbsibiu.ro, marcu.alex@yahoo.com

Abstract: - If in written or spoken form, language is an essence of human behavior. It is used for knowledge representation and transfer from one generation to another. Without language we wouldn't be able to any kind of communication. Today, we are all confronted by an unprecedented volume of information and most of it in text form. Using computer system to manage and access large volumes of information becomes a necessary evil for today, and even more for upcoming generations. System capable of understanding human language would significantly improve human-computer interaction. This paper presents an application framework for natural language processing in form of a ChatBot for the Romanian language.

Key-Words: - natural language processing, formal languages, parsing, natural language interface, computational model

1 Introduction

Since for many people, a large and growing fraction of work and leisure time is spent navigating and accessing the universe of information, classical computer languages and information querying methods are not an attractive and realistic option. Thus, the study of language has become a primary area of interest for science. Besides the vast amount of information that a system capable of understanding human language would have to access, it would first of all improve human-computer interaction.

Some years ago (in 2003), it was estimated that the annual production of books reached about 8 Terabytes. It would take a human being at least five years to read the new scientific material that is produced every 24 hours. Although these estimates where based on printed materials and do not include the increasingly amount of information produced electronically on the Web.

None of the current expert system can match the flexibility and accurate of a human conversation. And considering the level of ambiguity in some languages is amazing how human psychology has adapted.

Natural language processing (NLP) is defined [4] as “a field of computer science and linguistics concerned with the interactions between computers and human (natural) languages”. Computer system, understanding natural language, deal with machine reading comprehension, and represent a subtopic of NLP.

The application of natural language understanding described in these paper addresses text based processing. Researches have shown that this kind of processing applies successfully to: querying document from databases with desired topics, information extraction from documents ore messages, text translations from one language to another, question-answering system [1][3].

The system needs to participate actively in order to maintain a natural dialogue. Furthermore it requires verifying that things are understood and if not, an ability to generate clarification sub-dialogues.

Parsing input is more complex than the reverse process of output construction in natural language generation because of the potential occurrence of unknown and unexpected features in the input and the need to determine the appropriate syntactic and semantic schemes to apply to it.

The first popular program that uses natural language communication was ELIZA, developed by MIT beginning from 1960. Other common Chatbox ore Chatbot applications are Dr. Romulon based on the ALICE artificial intelligence chat platform and MathBot for answering simple number problems. A milestone in this application filed is the award-winning free natural language artificial intelligence chat robot A.L.I.C.E. (Artificial Linguistic Internet Computer Entity) using AIML (Artificial Intelligence Markup Language) [8]. A huge of chatterbots, chat bot, conversational agents and virtual agents from all over the World the Chatbots Directory [7].
There are two main motivations for developing computational models: the scientific motivation and the practical/technological motivation \[1]\[2]. The former one deals with obtaining a better understanding about how language works and tries to transverse complex theories as computer programs and then test them by observing how well they perform. The later, deals with the assumption that natural language processing capabilities may change the way computers are used today. Computers aware of natural language understanding could access and manage stored information in text form and in addition provide a user interface accessible to everyone.

In this paper we present the implementation of a ChatBot based on common concepts from formal language theory and natural language understanding. The program is implemented in Python programming language and may be obtained and accessed through the internet.

At the end we outline a set of extensions to the formal model used. These are based on our previous research on formal models for modeling and simulating dynamic systems.

Even if the model used does not faithfully match the way humans process languages, it is important only to produce the desired results.

2 Notions and terminology
The first step in making a computer capable of processing natural language is to define a set of rules that yield an exact communication needed for the computer, as contrary to a more ambiguous one accepted among humans. It is possible for a sentence to have any number of meanings even for a particular context. This raises a very particular problem for algorithms meant to understand human language, because computers programs are traditionally used in a very precise and exact way.

From formal language theory we know that a Chomsky generative grammar (shortly grammar) \[2], \[6], \[7], is a quadruple \(G=(V_N,V_T,S,P)\), where \(V_N\) and \(V_T\) are alphabets of nonterminal respectively terminal symbols; \(S \in V_N\) is the starting symbol or axiom and \(P\) is a finite set of pairs of nonterminal words from \((V_N \cup V_T)^*\), \(P=\{(u_i,v_i)\mid 1 \leq i \leq m\}\), so that any word \(u_i\) contains at least one nonterminal symbol. The pairs \((u_i,v_i)\) are called derivation rules or production rules or simple productions and will be denoted by \(u_i \rightarrow v_i\).

If the left hand side of a production rule consists of only a single nonterminal symbol, then we have a context-free grammar \[2]\[1]. The set of all sentences build in respect to the considered rules, is called language generated by the grammar, and formally defined as:

\[L(G)=\{p \in V_T^* \mid S \Rightarrow p\}\].

From the viewpoint of natural language processing, context free grammars are interesting for two reasons:
- the model is powerful enough to represent structures of natural languages;
- the model is simple enough to build efficient parsers to analyze sentences.

Having a model of a given language, the next step is to create an algorithm which tests a given sentence to see if it's well formed. Such type of algorithm is called parser. Technically, a parser or, more formally, syntactic analysis, is the process of analyzing a sequence of tokens to determine their grammatical structure with respect to a given formal grammar.

Every programming language has a parser and at least an interpreter to bring the code into a program. Where as for programming languages this task is considerably easy, the human languages are seemingly endless in complexity. A sentence in natural language may have several interpretations and choosing between them is related to the beliefs and knowledge of the person who communicates.

The following production rules are a simplified except from the grammar of Romanian language used in our application.

\[S \rightarrow PS\]
\[PS \rightarrow PREDICAT SUBIECT\]
\[PREDICAT \rightarrow PRONUME_INTEROGATIV VERB_COPULATIV\]
\[SUBIECT \rightarrow ARTICOL_NEHOTARAT SUBSTANTIV\]
\[PRONUME_INTEROGATIV \rightarrow 'ce'\]
\[VERB_COPULATIV \rightarrow 'este'\]
\[ARTICOL_NEHOTARAT \rightarrow 'o'\]
\[SUBSTANTIV \rightarrow 'masina'\]

According to these rules we can build sentences in a tree structure called derivation tree (e.g. fig. 1.).

![Figure 1. Derivation tree of a sentence.](image-url)
with the S symbol and attempts to populate the internal structure of the derivation tree to obtain a sequence of terminal symbols that matches the classes of the words in the input sentence. (b) bottom-up - attempts to populate the internal structure of the derivation tree starting from the sentence and searching for matching production rules that can stepwise lead finally to the top symbol S.

Simulating a system mainly supposes a model for experiment creation that matches best to the evolving of the real system and also supposes a set of processing procedures for this experiments that may indicate the optimal decisions for a further control of the system. The system simulation starts with initializing the system with data describing the initial state. The dynamics of the system consist in choosing the next activity, i.e. the next procedure to be executed.

3 Application description
The application represents an implementation of a Chatbot for Romanian language and is basically guided by the work and results from [1][2][5][6]. Although, it is an attempt to build a framework for easing up computer-human communication through natural language usage.

The user interface is simple and can easily be integrated in any web page. Figure 2 depicts the element of the interface, containing a text field for the user input, an optional button to send a sentence to the system and a list of conversation history.

![User Interface](image)

**Figure 2. User interface.**

The development is entirely in Phyten [17] as programming language and Django [18] for data models and web integration.

Functionally, the application supports two operation modes: (a) training mode – this mode is used to train the system by adding and classifying new knowledge; (b) conversation mode – this is the normal operation mode in which the system responds to the sentences passed by user.

Training data as well as testing data are collected from the Romanian language, due the underlying grammar.

In this first version the two operation modes are independent. We plan to extend this type of operation

![Diagram](image)

**Figure 3. Response generation process with internal stages.**
such that user input sentences may contribute to the knowledge of the system. A proposal for this is described in a later paragraph.

3.1 Internal structure and functionality
With respect to the concepts of grammars and parsers, we can now describe the composition and internal behavior of our system.

For a given input, in form of a sentence, we have to consider the following steps, in order to get an understanding from it. First of all the sentence has to be checked to see if it's well formed. Having passed this, it is then divided into three parts: the element that does the action, the action and the rest of the sentence containing extra parameters used to understand how and in which circumstances the action has been performed. Next a class is constructed which generates the action on the word performing it. A response according to the system's behavior is then provided.

Communication is the process of transferring information from one source to another. In learning mode, all of the information communicated to the system, will need in the beginning a few pre-coded actions, necessary for creating the initial links between parts of knowledge.

All words are, in essence, labels for something in real life and thus, they are the most basic elements of human communication. In common for all elements are the representation of their specifics/particularities and what are the actions that they can do.

A Word class contains two distinct elements: the parts and the action. With this representation it is now possible to create the sentence. Important here is to understand the way in which a word's meaning is affected in a certain sentence, and further more, the way in which the meaning of the sentence is affected by the phrase, and in the end, the general meaning affected by the context and the general

---

**Figure 4. Internal data representation and relations.**
topic of the discussion.

The other basic element of this system is the action which in this case is actual source code which will be invoked with the appropriate words contained in the sentence. The way in which this particular function is constructed will represent the way in which the sentence will be understood. For better performance a distinction is made between existing verbs. A verb that will expect to get an adverb will perform only a certain type of actions as opposed to a verb that expect an adjective. An even more deep separation can be made using a classification system. For example a verb can perform certain operations to certain types of words.

3.2 Integration and extension

For a future release of this application we consider two alternatives as extensions for the formal model of the grammar treating system. One based on fuzzy reasoning and the other on a stochastic approach. We present in the following some basic concepts for the stochastic version. A fuzzy modeling system is treated in an earlier paper [16].

Definition 3.1. A stochastic generative grammar (or shortly stochastic grammar) is a pair \((G, f_p)\) where:
- \(G=(V_N, V_T, S, P)\) is a Chomsky generative grammar and
- \(f_p: P \rightarrow [0,1]\) is a probability function having the property \(\Sigma(f_p(\alpha \rightarrow \alpha_i)) = 1\), where \(\alpha \rightarrow \alpha_i\) are all \(\alpha\)-productions from \(P\).

Extending the probability functions from productions to derivations we obtain the following results: \(f_p(S \Rightarrow q \Rightarrow r) = f_p(S \Rightarrow q). f_p(q \Rightarrow r)\), where \(f_p(q \Rightarrow r) = f_p(p)\), \(p\) being the set of productions applied.

Definition 3.2. The languages generated by the stochastic grammar \((G, f_p)\) is
\[ L(G, f_p) = \{p \mid p \in V_f' \text{ and } S \Rightarrow p, f_p(S \Rightarrow p) > 0\}. \]

Definition 3.3. A stochastic generative grammar of type \(i = 0,1,2,3\) is a pair \((G, f_p)\) where \(G=(V_N, V_T, S, P)\) is a Chomsky generative grammar of type \(i\) and \(f_p: P \rightarrow [0,1]\) is a probability function with the property \(\Sigma(f_p(\alpha \rightarrow \alpha_i)) = 1\), where \(\alpha \rightarrow \alpha_i\) are all \(\alpha\)-productions from \(P\).

Definition 3.4. A stimulation functions is a mapping \(f:[0,1]^n \times [1,2,\ldots,n] \rightarrow [0,1]^n\), \(f(x_1, x_2, \ldots, x_n, i) = (f_1(x_1, x_2, \ldots, x_n, i), \ldots, f_n(x_1, x_2, \ldots, x_n, i))\) verifying the properties:
- \(\sum_{i=1}^{n} f_i(x_1, x_2, \ldots, x_n, i) = \sum_{i=1}^{n} x_i \) \hspace{1cm} (1)
- \(f_i(x_1, x_2, \ldots, x_n, i) > x_i \) \hspace{1cm} (2)
- \(f_i(x_1, x_2, \ldots, x_n, i) < x_i \) \hspace{1cm} (3)

We notice that if \(\alpha \rightarrow \alpha_i, i = 1, \ldots, n\) are all the \(\alpha\)-productions from \(P\) then we can associate to them a finite probability field
\[ \mathcal{A}(\alpha) = \left( \alpha \rightarrow \alpha_1, \alpha \rightarrow \alpha_2, \ldots, \alpha \rightarrow \alpha_n \right) \]
(4)

On the set of these probability fields \(\mathcal{A}\) we define now an operator called stimulation operator.

A stimulation operator is a mapping \(E: \mathcal{A} \times [1,2,\ldots,n] \rightarrow \mathcal{A}\) defined as
\[ E(\mathcal{A}(\alpha), l) = \left( \frac{\alpha}{\alpha_1}, \frac{\alpha}{\alpha_2}, \ldots, \frac{\alpha}{\alpha_n} \right), \]
where \(q_i = f_i(x_1, x_2, \ldots, x_n, i), l \in \{1,2,\ldots,n\}\).

4 Conclusion and further work

At this early stage of development, our approach looks promising especially by the fact that “it’s working”. The modular development of the application framework enables us to experiment with further extension, as stated in paragraph 3.2.

One major concern for every system that has to interact with a human communicator is to deal with error validation. Furthermore, facing constantly such issues, it is necessarily to extend the dynamic system behavior with a learning strategy. An input validation would then use the parser for a preliminary check followed by the logical test. The later can only be done by understanding a sentence meaning.

A drawback that needs to be handled is the fact that the system is currently susceptible to assimilating bad information in the instruction mode.

Another issue we would like to investigate is the integration into an ecommerce web portal, where a potentially new client can ask questions about the products available, and this he way bypass tedious navigating through confusing menus.

References: