Abstract

At this moment in time many applications that work with voice recognition systems are begin created. These applications are exposed to different sources of background noise and also the people that operate them come from different backgrounds and have different accents. These facts can make a voice recognition system misinterpret its input. The method discussed in this paper uses Artificial Intelligence techniques to create an intelligent syntactic parser to process the input information to the applications that use voice recognition systems. In future research this parser will be used with a voice recognition system to create a general application that can be used by many other applications that need to process input through a voice recognition system.

1. Introduction

The main purpose of this research is to create a syntactic parser to process the input of a voice recognition system. Syntactic Parsers manipulate the declarative knowledge of the grammar to determine if a sentence is correct or not. Recognition is the process of identifying a string of words as syntactically well formed. Parsing associates a syntactic structure to those expressions that have been recognized. Recognition and parsing are processes that determine whether a particular sentence or a stream of words is a valid expression or not [Ortiz2000]. A Syntactic Parser is used in the application created by this research to get the input from a voice recognition system and filter out the background noise. Also it detects homophones (words that have the same sound but different meaning). A person can say the word “two” and the voice recognizer may detect “to”. The parser will verify the words using AI techniques to find out if the word inputted is a homophone of a syntactically correct word or if it is an error in the inputted word. Another feature of this syntactic parser is the fact that sometimes people tend to elongate a word. For example a person might say the word “tuuuurn” and this parser will compare the word with the words in the database and will assume that what the person means is the word “turn”.

This Syntactic parser is implemented using Definite Clause Grammars (DCG). Definite Clause Grammars are convenient ways to represent grammatical relationships for various parsing applications. DCG are an extension of context free grammars that have proven useful for describing natural and formal languages, and that may be conveniently expressed and executed in Prolog. A context-free grammar (CFG) is a set of recursive rewriting rules (or productions) used to generate patterns of strings. A DCG rule in Prolog is executable
because it is just a notational variant of a Prolog term that has the following general form: \( \text{Head} \rightarrow \text{Body} \).

### 2. Problem Definition

This Syntactic Parser verifies the definite clause grammar written in the database to prove that the command is syntactically correct. Then if the command is not correct the parser verifies each word on the command to prove if any of the following situations is happening. The first one works with the fact that people come from different backgrounds and have different accents. And therefore an application with a voice recognition system might misinterpret the input said by a person if he/she has an accent. For example, a person could say the word “\textit{two}” and the voice recognition system could recognize the word “\textit{to}”. The words “\textit{two}” and “\textit{to}” are homophones (words that have the same sound but different meaning). The syntactic parser solves this situation by verifying the word and if the word is not valid, but the homophone of the word is, the parser will replace the word with its homophone.

The second situation deals with the fact that people sometimes tend to elongate words. For example if a person says the word “\textit{turn}” but the voice recognition system corrected it to the word “\textit{turnnnn}”, the parser will look at the definite clause grammar for each of the words written in the database and it will eliminate each of the additional letters understood by the voice recognition system.

If the first two situations are not present, then the parser will eliminate the word because it will consider it background noise. This Syntactic Parser, also, calculates a certainty factor (CF). For the situation of the word being a homophone, the certainty factor is an empirical value. In the case of the elongation of words, the certainty factor is the rate of equal letters in the word inputted and the correct word.

Basically, what the syntactic parser does is that first it verifies if the command said is valid, if not it verifies each word on the command to prove if the homophone of the word is valid. If that option fails, then the parser will verify if the word said is an elongated version of the real word. If any of these conditions fail, it will skip the word because it is not valid.

This research uses as example the commands for Air Traffic Control. The database was created with the Definite Clause Grammar for these commands. This is an example of the DCG grammar for the air traffic controller’s commands.

```
grammar (Type,Homophone) -->
  idn_phrase (Type,Homophone),
  command_phrase (Type,Homophone).

idn_phrase (Type,Homophone) -->
  word (subj,Homophone),
  word (idn,Homophone),
  word (idn,Homophone),
  word (idn,Homophone).

idn_phrase (Type,Homophone) -->
  word (idn,Homophone),
  word (idn,Homophone),
  word (idn,Homophone).

command_phrase (Type,Homophone) -->
  word (cw,Homophone).

command_phrase (Type,Homophone) -->
  word (cw,Homophone),
  command_phrase (Type,Homophone).

command_phrase (Type,Homophone) -->
  word (mw,Homophone),
  word (unit,Homophone).

command_phrase (Type,Homophone) -->
  word (mw,Homophone),
  command_phrase (Type,Homophone).
```

3. Experimental Work

Examples of the output of the parser created using Prolog with the grammar for Air Traffic Controllers Commands are demonstrated on Figure 1.

![Output of the Intelligent Parser](image)

?- go.

Enter command to parse or q to quit.
DCG -->One six seven bahh begin descent.
Correct Command: one six seven begin descent

Enter command to parse or q to quit.
DCG -->One to to turn right.
Correct Command: one two (0.9) two (0.9) turn right

Enter command to parse or q to quit.
DCG -->Ones for five on course.
Correct Command: one (0.75) four (0.9) five on course

Enter command to parse or q to quit.
DCG -->q.

yes
?-

Figure 1: Output of the Intelligent Parser.

4. Conclusions

The results of this research have been successful with a small amount of commands in the database. More features can be added, and at the moment a research is being conducted to add more features and to build an application that uses a voice recognition system.

References


