Eugene Goostman the Bot

Vladimir Veselov, Ph.D.

Introduction

Mozhaysky Military Space Engineering Academy
Russia, Saint Petersburg

Baikonour Space Center, Kazakhstan

Eugene Goostman

- http://www.princetonai.com/bot
- Eugene was born in 2001
- The Loebner Prize finalist – 2001, 2002, 2003, 2005, 2008
- Team: Eugene Demchenko, Sergey Ulasen, Selena Semoushkina, Mikhael Gershkovich, Vladimir Veselov, Laurent Alquier (graphics)

Features

- Spell checker, typos corrector
- Advance pattern matching language
- Context sensitive pattern matching
- Random or defined responses
- Logical and math expression interpreter
- Constraint dialogs
- Dynamic tokens (loaded from database)
- Integration with external database(s)

Architecture

<Engine>
<Input>
<Variables>
<Actions>
<Language Processor>
<Input>
<MultiTokens>
<RegularMatchers>
<TreeMatchers>
<Themes>
<QAUnits>
<Patterns>
<Rules>
<ContextPatterns>
<Rules>
<Actions>
<Rules>
<Text>
<Output>
<Random>
<Output>
<Variables>
<Request>

Typos Corrector

abbout=about
abotu=about
abouta=about a
aboutit=about it
abscence=absence
accomodate=accomodate
accross=across
acheive=achieve
acheived=achieved
acheiving=achieving
acn=can
…

Pattern matching expressions

1. Token definition

<MultiToken name="qWhatBe">(tell (me|) (about|)|I am interested in|what (is|are|were|was|*ould be|)|know what)</MultiToken>

<MultiToken name="qWhatBeo">(#(qWhatBe)|)</MultiToken>

<MultiToken name="qWhoBe">(tell (me|) (about|)|I am interested in|who (is|are|was|were|*ould be|)|know who)</MultiToken>

<MultiToken name="qWhoBeo">(#(qWhoBe)|)</MultiToken>

2. Pattern Expressions

<Pattern>* **(#(qWhatAbout)|) (#(pYour)|#(pU)) name** *</Pattern>

<Pattern>* **tell me (#(pYour)|#(pU)) name** *</Pattern>

<Pattern>* **how * I call #(pU)** *</Pattern>

<Pattern>* **my name is * what * yours** *</Pattern>

<Pattern>* **do #(pU) have (a|) name** *</Pattern>

```
<Theme name="Mother">
 <QAUnit name="HaveMother" type="STATEMENT">
  <Pattern>* #(qDoYouHave) * #(mother) *</Pattern>
  <Pattern>* #(pYour) #(mother) * (well|feeling fine|fine) *</Pattern>
  <Pattern>mother</Pattern>
  <Random>
   <Text>Yes, of course I have mother! And I love her very much.</Text>
   <Text>My mother works on TV.</Text>
  </Random>
 </QAUnit>
<QAUnit name="MotherProfession" type="STATEMENT">
  <Pattern>* #(qWhoBeO) * #(pYour) #(mother) *</Pattern>
  <Pattern>* #(pYour) #(mother) #(qWhoBeO) *</Pattern>
  <Pattern>* (Does|if|whether|where) * #(pYour) #(mother) work* *</Pattern>
  <Pattern>* (Tell me (more|*)|) about #(pYour) #(mother) *</Pattern>
  <ContextPattern>* Who is she *</ContextPattern>
  <ContextPattern>* who she is *</ContextPattern>
  <ContextPattern>* (Does|if|whether|where) she work* *</ContextPattern>
  <ContextPattern>* what does she do *</ContextPattern>
  <ContextPattern>* #(qWhoBeO) #(pYour) #(mother) #(profession) *</ContextPattern>
<Random>
    <Text>My mother works on the Odessa TV and runs a popular program for teenagers "Speak Russian right!" Every
    Odessian teenager heard her famous "For those putzs who schmooze Russian in a goddam lame way: drop by and
    listen to mine!"</Text>
    <Text>If you lived in Odessa, you couldn't help but know my mom - she runs a popular educational TV program for
    teenagers about Russian language.</Text>
  </Random>
 </QAUnit>
```

Processing Math Expressions

```
<QAUnit name="Calc" type="STATEMENT">
  <Pattern>* (calculate|evaluate | tell me|what is|expression|) * #(MiddleNumber) (+|-
    |#(symStar)|#(symSlash)|plus|add|and|minus|subtract*|times|multipl* by|divid* by)
    #(MiddleNumber) *</Pattern>
  <Action>
    <Variable name="Arg1" type="String">=gettoken[TRACE_VARIABLE,1]</Variable>
    <Variable name="Arg1Num" type="String">=text.words2int[#(Arg1)]</Variable>
    <Variable name="SignStr" type="String">=gettoken[TRACE_VARIABLE,2]</Variable>
    <Variable name="Sign" type="String">=text.words2sign[#(SignStr)]</Variable>
    <Variable name="Arg2" type="String">=gettoken[TRACE_VARIABLE,3]</Variable>
    <Variable name="Arg2Num" type="String">=text.words2int[#(Arg2)]</Variable>
  </Action>
  <Action>
    <Text>
      <Expression>#(Arg1Num) #(Sign) #(Arg2Num)</Expression>
      <Variable name="Expr" type="String"/>
    </Text>
    <Variable name="Res" type="String">=text.evaluate[#(Expr)]</Variable>
    <Random>
      <Text>Am I a calculator for you? Well, I am... #(Expr)=#(Res).</Text>
      <Text>Well, just for you: #(Expr)=#(Res)... But hush - it's a secret information!!!</Text>
      <Text>Now I'm sure that you are one of those crappy robots from the Great Robots Cabal! And I know
    your password: #(Expr)=#(Res)! Now divide by zero and die!!!!</Text>
      <Text>#(Expr)=#(Res). Hope, in the nearest future, you'll try to estimate a square root of -1, and die of
    Overwhelming Imaginarity, crappy android!</Text>
    </Random>
  </Action>
</QAUnit>
```

Dynamic Tokens

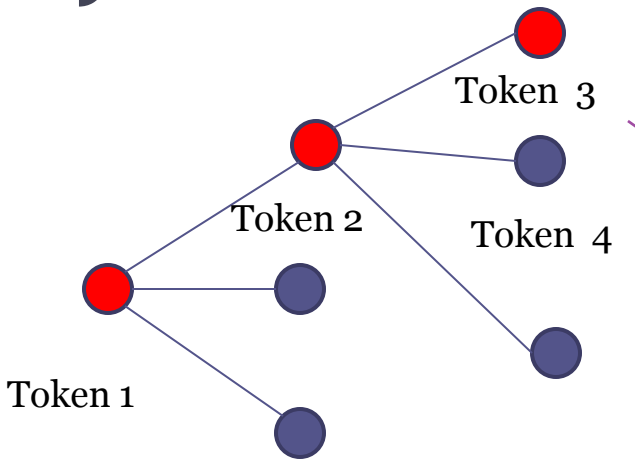
Problem: How to use a structured data in a chatbot?

Example: The information about countries stored in the database. Bot needs to answer the questions:
"What is the capital of <country name>?"
"What is the population of <country name>?"
etc.

Traditional approach will require long routine work to create patterns and answers for all countries.

How to answer: "Which country has more population: <country 1> or <country 2>?"

Dynamic Tokens



Tree Matcher

Database

Country	Capital	Population
France	Paris	50,000,000
Germany	Berlin	60,000,000
..		

Sentence: “<Token 1> <Token 2> <Token 3>”

Pattern: <Token 1> <Token 2> <Token 3>

Response: <text> <Token 3> <information>

Dynamic Tokens

1. Load tokens from the database.

```
<MultiToken name="dynCountries" type="DYNAMIC">
    SELECT CountryPattern FROM Countries
</MultiToken>
```

2. Define the pattern.

```
<Pattern>* (#(qWhatBe) | #(qWhatAbout) |) * (capital|capitol) * (of|in|) #(dynCountries)
    *</Pattern>
```

3.Extract the country name.

```
 <Variable name="country"
    type="String">=GetTokenByName[TRACE_VARIABLE,{dynCountries}]</Variable>
```

4. Retrieve the capital from the database.

```
<DataLookUp>
   <Table>
       <TableName>Countries</TableName>
       <Record><Field>Capital</Field><Variable name="capital“ type="String"/></Record>
   </Table>
   <Where>Country = #(country)</Where>
 </DataLookUp>
```

5. Generate response

```
<Text>Well, if you want it, the capital of #(country) is #(capital).</Text>
```

Logs

USER: what is a location of france
RESPONSE: Buy a map and hang it over your desk! France - country in W Europe, on the Atlantic and the Mediterranean Sea.

USER: what is its population
RESPONSE: Well, according to the latest secret data approximately 54.3 mln live in France.

USER: is it greater than germany
RESPONSE: I'd say that Germany is smaller than France.

USER: what is a size of germany
RESPONSE: Well, if you want to conquier Germany, you'll have lots of work - its square is about 356.8 thsd sq. km.

Benefits for Social Networks

- Chatbots can access user's information and provide comments on the user profile or on another users.
- Chatbots can use the information from news and events in conversation.
- Chatbots can access external data sources and use the information in conversation

Questions?

Thank you!