Chatterbot em AIML

Chatterbot implementado utilizando AIML para manutenção de diálogo sobre futebol

Bruno Moreno de Castro
Departamento de Ciência em Tecnologia
Universidade Federal de São Paulo, UNIFESP
São José dos Campos, São Paulo - Brazil
mcastrobruno@gmail.com

Wellington Azevedo Barros
Departamento de Ciência e Tecnologia
Universidade Federal de São Paulo
São José dos Campos, São Paulo – Brazil
uelito.sjc@gmail.com

***Abstract*—Este trabalho tem como objetivo a implementação de um chatterbot capaz de manter um diálogo sobre o tema futebol através de troca de mensagens de texto. A interface de usuário foi desenvolvida na plataforma web utilizando a tecnologia ASP.NET MVC para o front-end e C# para o back-end. A base de conhecimento do chatterbot foi implementada utilizando o padrão AIML.**

Inteligência Artificial, Chatterbot, Aprendizado Supervisionado, AIML, Program#.

I. INTRODUÇÃO

Este trabalho propôs a criação de um chatterbot responsável por responder perguntas relacionadas ao futebol. Segundo Simon Laven (LAVEN, 2002), um chatterbot é um programa que tem por finalidade simular uma conversação escrita, com o objetivo de, pelo menos temporariamente, enganar um ser humano induzindo-o a pensar que está falando com outra pessoa. Este sistema deverá possuir uma interface Web para que o usuário possa iniciar e manter uma conversa com o chatterbot, assim como configurar e treinar sua base de conhecimento.

Para tanto, foi utilizada a linguagem AIML (Artificial Intelligence Markup Language). De acordo com Wallace (2001), a AIML descreve uma classe de objetos de dados chamados objetos AIML e parcialmente descreve o comportamento dos programas que os processam. Estes objetos são constituídos de tópicos e categorias. Por sua natureza, objetos AIML estão de acordo com documentos XML (Extensible Markup Language). Como o XML é uma forma restrita de SGML, Standard Generalized Markup Language (ISO 8879), objetos AIML também estão de acordo com documentos SGML. Segundo Flynn (2003), XML, ou linguagem de marcação extensível, é feito para aumentar a funcionalidade da Web fornecendo uma identificação mais flexível e adaptável para informações. É chamado extensível porque não é fixa como HTML (HyperText Markup Language, uma linguagem de marcação simples e pré-definida). Pelo contrário, XML é na verdade uma "meta-linguagem" usada para descrever outras linguagens, que permite desenvolver linguagens de marcação próprias e customizadas para infinitos tipos diferentes de documentos.

De acordo com Silva (2003), existem diversos tipos e categorias de chatterbots conforme sua finalidade: Entretenimento: são aqueles que têm por finalidade divertir o usuário. Para tanto, é preciso que o chatterbot simule melhor a personalidade humana, com gostos e desejos próprios; Chatterbots FAQs: têm a mesma finalidade que os FAQS (Frequently Asked Questions, ou perguntas feitas com freqüência), porém usando linguagem natural para se comunicar com o usuário; Suporte ao consumidor: são chatterbots que ajudam o cliente a descobrir e fornecer a solução para um determinado problema; Marketing: são os chatterbots responsáveis por fazer a propaganda de determinado produto ou serviço; Chatterbots de propósito geral: são aqueles sem um objetivo definido. Alguns destes chatterbots têm a capacidade de aprender o que os usuários os ensinam.

Cada chatterbot baseado em AIML possui um módulo chamado interpretador AIML, que é responsável por identificar a entrada do usuário na base e retornar a resposta adequada. Estes interpretadores podem ser desenvolvidos em diversas linguagens e ambientes.

Segundo A.L.I.C.E. (2003), entre os interpretadores mais populares estão:

Program Z: é implementado em Common Lisp. Esta implementação vem junto com completa hospedagem grátis e ferramentas de desenvolvimento AIML no site www.pandorabots.com;

Program D: é uma implementação em Java. Esta é a versão que usa a tecnologia mais moderna e com compatibilidade total com o padrão AIML;

Program M: é implementado em uma linguagem antiga chamada SETL (Set Theory and Mathematical Logic). É mais uma especificação formal para AIML do que uma aplicação prática e funcional. Para usar o Program M, é necessário instalar a linguagem SETL, que roda somente em máquinas com Linux;

Program E: (mais conhecido como "PHiliP") é uma implementação em PHP que está gerando muita expectativa. Ainda está em seus estágios iniciais, porém é compatível com o padrão AIML e possui uma comunidade que está crescendo rapidamente;

Program V: é uma implementação em Perl de um interpretador AIML e também é compatível com AIML 1.0.1.;

Program P: mais conhecido como PASCALice, tendo sido desenvolvido em Delphi.

Program#: interpretador mais conhecido para ser utilizado sobre a plataforma .net.

Dentre estes interpretadores, o Program # (PROGRAM#, 2012) parece ser mais adequado às necessidades do projeto.

II. MOTIVAÇÃO

Os chatterbots existem há muito tempo porém o interesse desta ferramenta em certos negócios é um atrativo e um desafio nos tempos de internet 2.0 onde a revolução de usuários e dados conectados é muito grande.

Com a internet os chatterbots ficou acessível ao público em geral e ao mesmo tempo criou para os web sites a necessidade de terem uma interface parecida com a humana para fornecer informações de forma agradável à milhares de pessoas por dia, fazendo diversas interações e/ou ações como um chatterbot que caça pedófilos na internet isso é um avanço e tanto em termos de um robô conseguir encontrar criminosos online apenas conversando.

Embora seja um começo muito promissor o fato é que mais e mais esforços estão sendo direcionados à pesquisa e desenvolvimento de chatterbots. Entretanto adquirir inteligência humana ainda seja um sonho distante, a tecnologia dos chatterbots só tende a crescer, parece certo que chatterbots têm um papel significante na internet do futuro.

III. OBJETIVOS

Afim de projetar um chatterbot que atende o mínimo de requisitos para uma conversa simples ao passo que internamente sua estrutura esteja bem avançada para que o chatterbot possa aprender com o "professor" que lhe ensinará de forma segura e correta questões que envolve o mundo do futebol. As implementações da base de conhecimento serão feitas com AIML tendo o foco de explorar todas as tags possíveis para construir um bot bem dinâmico.

O aprendizado será supervisionado, ou seja, terá um "professor" que lhe ensinará como aprender de forma muito fácil, apenas ira indicar ao chetterbot que aquela conversa é para ele aprender e gravar em sua base de conhecimento o tornando mais abrangente e rico em conhecimento especifico com tema de futebol.

Este sistema terá uma conversação simples e direta com respostas corretas e criativas, já que ele terá um instrutor para enriquecer seu conhecimento, ao final o chatterbot irá reconhecer com quem conversará com ele, responder perguntas especificas, reter conhecimento e etc.

IV. METODOLOGIA EXPERIMENTAL

Os métodos estudados e experimentados foram de linguagem de programação com seus respectivos programas compatíveis com o AIML, em testes feitos escolhemos o Program # (PROGRAM#, 2012) que pareceu ser mais adequado às necessidades do projeto e as nossas abilidades.

O desenvolvimento da base de conhecimento iniciou-se pela comparação e metodos utilizados da base de conhecimento padrão do chatterbot A.L.I.C.E. Esta base é de distribuição gratuita e está disponível em diversas línguas. Entretanto, ainda não existe uma base AIML gratuita disponível em português. Conforme fomos avançando no projeto tivemos dificuldades durante o precesso de implementação já que todas bases complexas são em ingles e devido às diferenças entre o inglês e o português, algumas categorias não foram traduzidas e, por consequência, retiradas da base de conhecimento. Isto deve-se ao fato de que algumas expressões só existem em inglês, e não em português, além do fato de que existem palavras que são sinônimos em inglês mas que não existem correspondentes na língua portuguesa.

Abaixo vamos citar algumas tags que foram implementadas no nosso projeto e como essas tags se comportam emu ma linguagem AIML.

A. *<aiml> Tag*

A Tabela 1 mostra o código AIML que ilustra uma utilização do <aiml>, cada linha de arquivo .aiml começa com a tag <aiml> e termina com </aiml>.

Tabela 1. <aiml> Tag

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <aiml>
3     <category>
4        <pattern> OLA BOT </pattern>
5           <template>
6              Ola meu amigo!
7           </template>
8     </category>
9  </aiml>
```

Exemplo de código AIML.

B. *<category> Tag*

São chamados de unidades básicas de um diálogo, cada categoria é uma unidade fundamental de conhecimento no chatterbot. Uma categoria consisti em uma entrada do utilizador, sob a forma de uma frase, uma resposta a entrada do usuário, e um contexto opcional. As categorias são organizadas por assuntos e armazenados em arquivos com a extensão . aiml, afim de facilitar o conhecimento da base e seu progresso de manutenção.

C. *<pattern> Tag*

Nesta tag contém uma possível entrada do usuário, há apenas um único <pattern> em uma tag <category> e deve ser o primeiro elemento a ser definido. Além disso, as palavras são separadas por espaços individuais, e curingas "*" pode ser substituída por uma parte da frase no início ou no fim.

D. *<template> Tag*

A tag <template> contém possíveis respostas do chatterbot para o usuário, ela deve estar dentro da tag <category> e ser colocado logo após o <pattern>. A maior parte do chatterbot à informação é limitada por este elemento. Essa marca pode salvar os dados e ativar outros programas, ou até mesmo dar

respostas condicionais e chamar outras categorias. No código AIML apresentado na Tabela 1, o <template> fornece uma resposta para o usuário, que neste caso apenas exibe a frase "Olá meu amigo!", como mostrado nas linhas 4-6.

E. <star index = "n"/> Tag

A tag <star index = "n"/> captura um fragmento particular, contida na sentença do usuário. O índice n indica o componente frase que vai ser mapeado e capturado, assim observa-se que:

- <star index="1"/>: equivale ao primeiro fragmento do texto.
- <star index="2"/>: equivale ao segundo fragmento do texto.
- <star index="3"/>: equivale ao terceiro fragmento do texto.

Com este comando é possível armazenar fragmentos de texto do usuário, o atributo n no índice = "n" é opcional e se for omitido o valor 1 é assumido, sendo assim a tag <star/> é o mesmo que <star ="1"/>.

Tabela 2. <star> Tag

```
1   <category>
2     <pattern> EU SOU * </pattern>
3     <template>
4       Eu sou <star/> também.
5     </template>
6   </category>
7   <category>
8     <pattern> UMA * É UMA * </pattern>
9     <template>
10      Quando uma <star index="1" /> não é uma <star
    index="2" />?
11    </template>
12  </category>
```

Exemplo de código AIML.

A Tabela 2 exemplifica o uso da tag <star> no código linha 8 a 13. Correspondendo o trecho do dialogo genérico:

Usuário: UMA * É UMA *

Bot: Quando uma * não é uma *?

A partir deste modelo de interação genérico, o seguinte dialogo pode ser alcançado através desta tag.

Usuário: Uma rosa é uma flor

Bot: Quando uma rosa não é uma flor?

Na linha 9, quando uma possível entrada do usuário é definida, dois index são armazenados, o primeiro curinga identifica "rosa" e o segundo identifica a palavra "flor".

F. <srai> Tag

Uma das propriedades mais uteis do idioma AIML é a possibilidade de visar diferentes <pattern> de entrada do usuário para um único <template>. Assim, o interpretador do AIML pode eficientemente procurar uma resposta a partir de diferentes tipos de entrada do usuário. Esta possibilidade é obtida usando a tag <srai>.

Estas tags são as mais usadas em AIML existem outras que também são de suma importância mas para não deixar o texto muito longo não vamos cita-las aqui. Nossa metodologia se baseou também em um editor AIML que ajuda muito a criar iterações de conversar e também é muito útil em termos de testes com a lógica aplicada no código, usando o editor AIML o código pode ser depurado e melhorado. Usamos o editor GaitoBot que é uma ferramenta off-line, ele é implementado em linguagem C# e é muito aconselhado para testar os código em AIML.

V. RESULTADOS/DISCUSSÕES

A primeira etapa do projeto foi a configuração do programa interpretador AIML. O interpretador usado no projeto foi o Program #, pela sua total compatibilidade com o padrão AIML e sua compatibilidade com o Sistema Operacional windows já que este foi escrito em C#. O editor AIML GaitoBot também usado no projeto tem compatibilidade total com o windows e seu pré-requisito é no mínimo a versão 4 do .net. A segunda etapa foi a criação de uma base de conhecimento AIML. Esta base se encontra na pasta aiml do projeto e contém dois arquivos AI.aiml e learn.aiml.

Nos testes iniciais, o chatterbot foi meio confuso e a maioria das perguntas não era identificada corretamente. Muitas vezes a categoria mais geral AI.aiml era acionada e respondia que não sabe a resposta e isso queria dizer que o bot precisa de mais treinamento ou seja um "prefessor" teria que ensina-lo a responder aquela pergunta. O programa muitas vezes se confunde com algumas perguntas e outras não tendo em visto que é um trabalho inicial e especifico em um assunto tema futebol. Com o processo de aprendizagem o chatterbot ficou muito mais eficiente conseguindo aprender e responder perguntas ensinadas a ele.

VI. CONCLUSÕES

O projeto de Inteligência Artificial reuniu conhecimentos sobre processamento de linguagem natural, XML e o editor da linguagem AIML. O objetivo foi tentar criar um programa de computador que pensasse como um ser humano no mundo do futebol como se fosse um "wiki".

Para a implementação do programa, foi usada a linguagem de marcação AIML, que além de poderosa é fácil de ser usada. É importante ressaltar a dificuldade encontrada em se conseguir desenvolver uma base de conhecimentos padrão no idioma português, como a própria gramatica não é tão simples regras e mais regras nos limitamos a atender ao menos o básico da linguagem portuguesa.

Ao analisar o esforço dedicado ao processo de criação da base de conhecimento, percebemos que se começar com a aprendizagem do bot primeiramente isso nos teria feio um ganho e tanto com a base de conhecimento mais solida e cheia de perguntas concretas do futebol. Por esses motivos a base de futebol ficou relativamente pequena, contendo basicamente informações sobre alguns jogadores e alguns campeonatos.

Outro ponto interessante do projeto foi notar que, por sua própria natureza e funcionamento, o Program # não permite que o chatterbot seja pró-ativo, ou seja, que faça suas próprias perguntas. É possível, entretanto, que a maneira como as perguntas são respondidas provoquem mais questionamentos. As conversas onde o chatterbot simplemente responde às perguntas tendem a se tornar curtas e pouco interessantes.

Como recomendação para trabalhos futuros seria melhorar a base de conhecimento. E também nas pesquisas que nós fizemos tem a possibilidade de um chatterbot emitir áudio e nós emitirmos áudio também, ou seja, olhando para o futuro poderemos conversar abertamente com o programa e isso junto com bibliotecas já existentes e aberta ao público levaria o conceito de chatterbot a um estágio de desenvolvimento muito superior ao que temos hoje em dia. Existem outras linguagens XML para o desenvolvimento de chatterbots mais complexas que não foram abordadas neste projeto e que levam em conta alguns pontos negativos da linguagem AIML como, por exemplo, a dificuldade de se determinar o contexto da conversa. Também seria interessante o desenvolvimento de um programa interpretador AIML próprio em alguma outra linguagem de programação com mais recursos e documentação do que tem hoje.

Referências

[1] Wallace, Richard (2003). Artificial Intelligence Markup Language (AIML) Version 1.0.1. Disponível em http://www.alicebot.org/TR/2001/WD-aiml/ Acessado em 25 de Maio de 2015.

[2] AliceBot (2015), AliceBot, Disponível em: http://alicebot.blogspot.com.br, Acessado em 25 de Maio de 2015.

[3] GaitoBot (2015), Aiml Chatbot Editor, Disponível em: http://www.gaitobot.de/gaitobot/, Acessado em 05 de Junho de 2015.

[4] Program# (2015), Download Program# 2.0, Disponível em: http://aimlbot.sourceforge.net, Acessado em 25 de Maio de 2015.

[5] Template utilizado em publicações IEEE, Disponível em: https://www.ieee.org/conferences_events/conferences/publishing/templates.html, Acessado em 20 de Junho de 2015.