Computer Science • 13 (2) 2012 http://dx.doi.org/10.7494/csci.2012.13.2.123

Fernando A. Mikic
Juan C. Burguillo
Ana Peleteiro
Marta Rey-López

USING TAGS IN AN AIML-BASED CHATTERBOT TO IMPROVE ITS KNOWLEDGE

Abstract *Nowadays, it is common to find on the Internet different conversational robots which interact with users simulating a natural language conversation. Among them, we can emphasize the chatterbots based on AIML language. In this paper we present an AIML based chatterbot that shows as its main contribution the use of tags and folksonomies. Thanks to its use, we can generate a context for each conversation, being able to maintain a state for each user in the system, and improving the adaptation capabilities of the bot.*

Keywords AIML, chatterbots, folksonomies, tagging, natural language

1. Introduction

Nowadays, it is becoming frequent that we find different types of conversational robots, commonly known as chatterbots or bots [7] in different areas of communication technologies, particularly on the Internet. The use of chatterbots offers the user a closer experience to interact with it through natural language, in a way that the user thinks that he is speaking with a human. However, there is still a long way to make the user feel as if he is relating to a human, and not to a robot.

One of the biggest problems that the chatterbot shows is randomness when choosing among several possible answers to a single question, without leading the conversation to any context. This situation results sometimes in inconsistent answers to the question asked that leads to a lack of meaning in conversation that will surely be detected by the user. With the idea of improving these systems, we have developed TagBot. We try to solve the problem of proper context in a conversation by using tags that allow the bot to maintain a state that can define the context of the ongoing conversation. The main contribution of this paper is a model of using tags and folksonomies for providing light semantics to chatterbots, i.e., to allow it to select the most appropriate conversations depending on the context. This model has been implemented in a new interpreter, denoted as program G, and a new chatterbot, denoted as TagBot from now on.

In this article we focus on the description of our TagBot, and in particular regarding its new tagging model and interpreter. The rest of the paper is organized as follows: in Sect. 2 we take a brief look at the key technologies used in the development of TagBot. In Sect. 3 we focus on its architecture. In Sect. 4 and Sect. 5 we show the characteristics of TagBot, especially the tagging system. Finally, we present our conclusions and the future work planned.

2. Technologies used

There were two main basic technologies for the development of our TagBot. Firstly, AIML language [15] that has been selected for its wide acceptance within the developer community, as well as our own experience in the development of these bots with that language. Secondly, we have the tagging which has become important in the last years, thanks to the social characteristics and semantic possibilities it offers. In the following sections we make a brief review of these two technologies.

2.1. Chatterbots and AIML

A chatterbot is a type of conversational agent, i.e., a computer program designed to simulate an intelligent conversation. It processes users' inputs in natural language and it looks up in its knowledge base to return an answer that imitates the human's one. Some of the most relevant chatterbots nowadays may be (more information on different models can be found in `http://chatbots.org`):

- ALICE (Artificial Linguistic Internet Computer Entity) [2]: One of the most ground-breaking projects in the field of Artificial Intelligence. ALICE is the project developed with AIML (Artificial Intelligence Markup Language), used to develop software chatterbots.
- Cleverbot: [3] A chatterbot that has as a basic learning strategy to repeat what others have said to it. The objective of this strategy is to have such a complex dialogue database that it can talk about anything with anyone.
- Ultra Hal [5]: A chatterbot that acts as a personal assistant, calendar, etc. (including Facebook). Hal is capable of discussing any topic and learning from users. Hal's personality is dynamically modified to be just like the users' closest friends, based on what these friends have taught to Hal before, while ignoring things learned from strangers.
- Chip Vivant [14]: This is another type of chatterbot whose objective is to answer basic, commonsense questions. It uses simple deductive reasoning, instead of having a massive responses database.
- FreeHAL [4]: This application uses semantic networks and employs pattern recognition, stemming, part-of-speech databases, and hidden Markov models in order to best mimic human behavior in conversations. Its main characteristic is that FreeHAL is able to add its own knowledge (the program expands its knowledge base through typed communication with the user).

One of the most common technologies used in developing chatterbots is the language AIML (chosen to develop our system). It is widely used in the development of software agents that communicate with their users in natural language, being its greatest exponent the previously cited ALICE bot.

The AIML language was developed by Richard Wallace and the Alicebot open source community between 1995 and 2000. AIML produces XML text files with a specific structure, that constitutes the knowledge base of the chatterbot. The "categories" are the fundamental knowledge basis, and they consist of at least two elements: the "pattern" and the "template".

In general, the performance of AIML is based on a stimulus-response model, in which the stimulus (the user's input) corresponds to the "pattern", and the response that the chatterbot shows to the user is its associated "template" (Fig. 1). All these actions, i.e., looking for the adequate pattern and showing the related template, are carried out by a data treatment engine, existing many different versions (Program D, Program E, etc.).

2.2. Tagging

In the last few years, Web 2.0 technologies have achieved great acceptance among users, and specifically tagging is becoming a popular practice.

We can consider tagging as a distributed process where resources or objects are described or characterized by tags (labels or sets of terms in natural language). The aggregate result of this process is called folksonomy [12]. This term was coined by

```
<category>
    <pattern>
        <!-- possible user's input -->
    </pattern>
    <template>
        <!-- chatterbot response-->
    </template>
</category>
```

Figure 1. AIML example

Vander Wal in 2005, [12], and it is defined as a mechanism that describes existing resources using the vocabulary of people.

The value of folksonomy is that users can use their own vocabulary to explicitly provide added value to the content they are consuming, both in their capacity as users and producers. One important aspect is that folksonomy consists of terms in a flat namespace, i.e., no hierarchy and no directly specified family relationships (it is considered, however, that tags are related to each other). In folksonomies, representation is free and the user is advised to assign multiple tags to each element to classify it. There are neither fixed nor hierarchical categories. Thus, a folksonomy is simply set of terms that have been used to label contents, where there are neither predetermined sets of terms nor labels for classification.

Folksonomies offer users a high degree of freedom to describe information and objects from their point of view and in their own words, providing background information of their experience with them, their knowledge, and so on.

Tagging is a compromise between a description and a categorization. Thus, a search could be enriched with the knowledge underlying the folksonomy, where there are semantic relationships between tags (analysis of co-occurrence), degrees of relevance, and even resource potential interests.

Nowadays, folksonomies are widely used in a extensive kind of systems and for numerous purposes. Only as some examples we can cite: creation of semantic metadata for the right description of resources [1], improving sharing information through peer to peer networks [16], personalized recommendations in social systems [9], characterization of users and their behavior [13], uses in on line public library catalogs [10], relations among users searching in their profiles [11], information structuring [6], etc.

In our case we have a system where the resources are the categories within the AIML bot brain, where the tagging is performed on those resources, but not directly. The programmer initially labels the basic categories introduced in memory, used to initialize the system. However, once the initial memory is created, the users come into play, being able to modify the tags by eliminating or increasing their importance.

3. TagBot architecture

Our project consists of a set of Java files divided into different packages. It also includes configuration files, images that are part of the GUI, and a properties file that is created for easy modification of some parameters of the system. Another aspect to consider in the design of our application is the user interface that is based on packages Java Swing and Java AWT.

From a high degree of abstraction, we can consider the TagBot divided into two main parts: first, its knowledge base and second, our new interpreter, named Program G that provides the support for AIML with tags.

The knowledge of the bot is represented in the knowledge base that is simply a database. In our particular case, being based on AIML, the chatterbot follows the categories pattern previously presented. That is, the AIML knowledge base has the structure of a pattern-matching system and the categories of knowledge are stored there. These categories are composed by patterns (to be matched with user input) and templates (containing the system response).

The interpreter is based on an existing Java project named Program D [8], created and developed by Noel Bush and a large group of collaborators. We have modified and improved this Program D taking into account concepts about tagging to maintain the user state in the conversation through the use of tags. Program G acts as a link between the user and the knowledge base of the bot. The architecture of the interpreter can be divided into three levels (Fig. 2):

1. GUI (interface): The part of the bot where the user input is registered, and where the responses generated by the bot are shown.
2. Interpreter: Responsible for performing input processing and for the construction of the response. A tree of categories of knowledge is created and stored in the knowledge base of the bots, in this way, they can process the statement of user input and seek for the most appropriate category for an answer. We developed a new interpreter (Program G) that provides support for tags and folksonomies within AIML.
3. Functionalities: They are specifically responsible for providing certain services, like interacting with the operating system, searching the Internet or communicating with other bots.

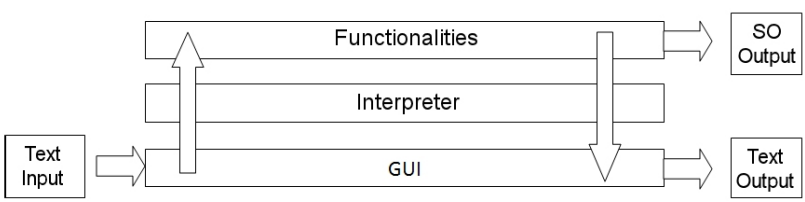

Figure 2. Levels of the system

As mentioned earlier, due to the magnitude of the project, it presents a modular approach, being composed of a set of Java files divided into several modules, each one with a very specific function, such as: instantiation of variables system-wide, the obtaining of configuration properties for the labeling system, loading AIML files on the basis of knowledge, response processing (performing system calls if necessary), changes in properties of a bot, initialization of input and output interfaces, gathering information from the responses received, etc. The next section discusses the most important modules in detail.

4. TagBot features

The main objective achieved with regard to the present work is to develop a bot that provides some added value to the current models. The TagBot maintains and uses the information it has about the user and the context of the conversation.

To do this, we have to provide our bot with the ability to learn about the user and also with solutions to get information from the network. In short, we have developed a chatterbot that evolves according to the wishes and the interaction with the user (Fig. 3).

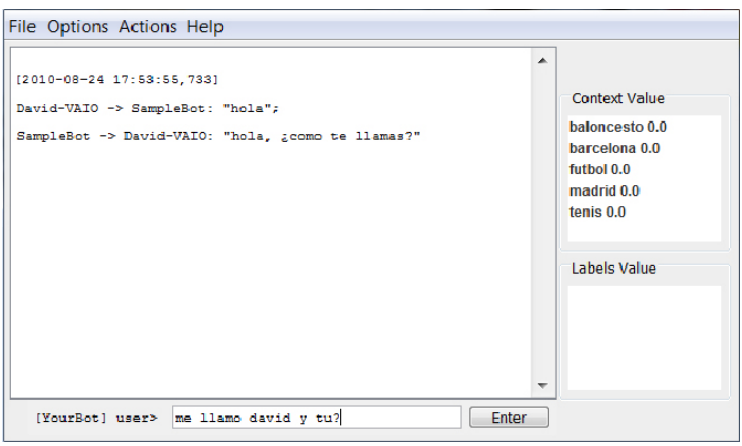

Figure 3. Sample conversation with TagBot

We have created a new bot adapted to every user that behaves as his personal assistant or avatar, with semantic features that connect tagged conversations. To do this, we have made changes to the AIML markup language, adding new fields to attributes, and pointers to other parts of the state graph.

In this section we provide a description of the modules we have generated for the TagBot in the interpreter, as well as in the core functionality and the interface module.

4.1. Modification of AIML

The main difference between our chatterbot and others is the tagging that allows for different types of users to define a set of categories using metadata, providing more information about them.

We have adapted the existing AIML, adding two new attributes for the element category that allows us to define them more appropriately (Fig. 4). Besides, they permit us to establish levels of differentiation within the categories:

- For the first attribute we have upgraded the category tag (to add metadata to each category) in this way: *<category labels="(name value)">* (there are no restrictions beyond compliance with the format *<name, value>*, being *name* a string and *value*, a numeric item).
- The second attribute is based on the idea of making a link between different terminal nodes of the system. In this way we can obtain information about the category that can be accessed next. This change is made adding a new attribute to the tag category. The new attribute is of the following type: *pattern:that:topic:Bot* and it is composed of a series of pairs *address* and *number of access*.

```
<?xml version="1.0" encoding="UTF-8"?>

<aiml versión="1.0.1" xmlns="http://alicebot.org/2001/AIML-1.0.1"
xmlns:html="http://www.w3.org/1999/xhtml"
xmlns:xsi="http://www.w3.org/2001XMLNSchema-instance"
xsi:schemaLocation="http://alicebot.org/2001/AIML-1.0.1 http://aitools.org/aiml/schema/AIML.xsd">

<topic name="saludos">

<category labels="inicio 10.0"><pattern>hola</pattern><template>hola, ¿cómo te llamas?</template></category>

<category link="NOMBRE : * : SALUDOS : SampleBot;2; "><pattern>me llamo
_</pattern><template><srai>Nombre</srai>/template></category>

<category link="HOLA : * : SALUDOS : SampleBot;1;"><pattern>Nombre</pattern><template>encantada de conocerte, mi nombre es
SampleBot</template></category>

<category><pattern>Lo suficiente. Sé que me gusta.</pattern><template>hola</template></category>

<category><pattern>hola, ¿cómo te llamas?</pattern><template>me llamo David</template></category>

</topic>

</aiml>
```

Figure 4. New AIML attributes

With this, we create a base to work to obtain information about categories and user behavior.

4.2. System tag clouds

With the new features explained in the previous subsection, we were able to include tagging within the system, thereby facilitating the selection of the adequate category. For this, we have a set of tags, that allow us to refine the knowledge provided by the user and stored by the chatterbot.

In our system we have considered four tag clouds. A tag cloud is a set of tags where the weight or graphical size of them is bigger for those that are more used. Each of these four tag clouds serves to define an element of the conversation, i.e., the bots try to give sense to the context, making a contextualization of the ongoing conversation, or simply allowing the TagBot to better understand the user and to select an answer adapted to their needs:

1. Topics tag cloud: We define a tag cloud with all the existing topics in the database, meaning as topics all the issues related with conversations that the bot has knowledge about. The weight of this tag cloud depends on the number of categories that the bot has. Thus, this tag cloud and its related weights evolve with the new categories stored by the bot in its knowledge base when chatting with the user about new topics.
2. Context tag cloud: Through this tag cloud, we contextualize the conversation, giving more or less weight to most recent tags, and decreasing the value of the tags that have not been activated recently.
3. User preferences tag cloud: The user is able to easily modify this tag cloud, since he/she is the one who chooses the preferences among the different topics. In addition, this tag cloud is modified in terms of access to different categories and it defines the user's intentions at the time of the conversation, i.e., what topics are more often chosen by him/her.
4. Bot preferences tag cloud: This tag cloud is introduced by the botmaster and defines the interests of the bot when bringing a topic of conversation. As a difference with the three previous tag clouds, this one can only be modified accessing to a configuration file.

5. Tagging system operation

We want the bot to constantly learn from the user. For this, relationships between external stimuli introduced by him/her and bot responses have to be established.

5.1. Evolution of the user preferences tag cloud

In the system we have defined, the tag cloud that has more weight is the one that defines the user, that allows the bot to foresee the most appropriate topic of conversation to continue the communication process.

As we mentioned earlier, every user is associated with a different tag cloud where each tag has a weight, loaded in the moment of initialization; so the bot can know in advance what categories are more interesting for chatting with the user. This tag cloud can be initially configured by the user, but it evolves dynamically following user dialogue with the chatterbot.

Thus, if a category has more than one possible answer, the bot automatically performs a selection process based on user preferences. This process involves calculating the distance from the user's tag cloud to the tag cloud of all the answers belonging to

the category, i.e., the calculation is based on the distance between the two clouds, the one associated with a response and the one associated with the user, being selected the one that most closely matches the user's needs.

To calculate this distance, we use the following method (that is used in all calculations of the distances of tagging in the system): we note W_c as the weight of the metadata tag cloud belonging to a category, W_u the weight of the same tag but associated with the cloud of user preferences, and R_j the value of Eq. 1 for each answer j of the associated category.

$$R_j = \max\left(\sum_{i=1}^{n} W_c \cdot W_u\right) \tag{1}$$

The chosen answer is the one that has associated greater R_j value, and in case of equality, there is a random choice.

In addition, we have designed and implemented a method of tag propagation, that refines the model of the user's preferences naturally and automatically along the conversation. This system consists in modifying the user preferences tag cloud in terms of the tags that appear linked to the user responses during the conversation.

Finally, just to mention that, since it is the user the one who decides on what topics he wants to talk about, we have also considered the possibility of modifying the tags in both text and graphical mode.

5.2. Conversation context

We also have designed and implemented a tag cloud to model the present state of the conversation exchange between the user and the bot. The creation of this context tag cloud takes place at the time of system initialization, when its tag weights are set to zero. We do not save context information between executions, as we assume relatively independent conversations among different sessions. Tags on this context are added during the conversation to model what they are chatting about, gradually modifying the tag cloud. When a response activates a specific category that has a tag itself, this tag weight is increased in the context tag cloud in a particular factor. Furthermore, we have introduced an attenuation factor that is applied to all the tags of this tag cloud in every interaction. Therefore, only frequent and recent tags are highlighted fitting the context of the ongoing conversation.

Incremental and attenuation factors are configured by the user, allowing this way as an adjustment between the calculation of a context closer or further away depending on those configuration parameters. That is, if the difference between these two parameters is reduced we model a closer context, temporarily speaking, while large difference models a long one.

6. Conclusions and future work

In this paper we have presented TagBot, a conversational bot that is able to learn from the user and its environment by means of tagging. We have focused mainly on the tagging part, and we have also presented Program G, our new interpreter to support AIML with tagging. Throughout tagging, we have obtained a better definition of the user, the chatterbot itself, and the ongoing conversation. Besides, it also provided us with the necessary means for the development, learning and evolution of TagBot.

We designed and implemented several tag clouds that allow us to contextualize the user, the conversation, and the bot, providing this last one with an increased capacity of decision to get a useful answer. The spread of tags, its choice, and the choice of the best categories at each time has been possible by developing several techniques that interact among the tags and with the bot learning ability.

A major problem of chatter bots is their isolation when facing knowledge that is not in its knowledge base, so the bot cannot have a conversation regarding that topic. Already as a future line, we will try to alleviate this problem by allowing bots to communicate among them in order to enhance their knowledge.

Acknowledgements

We want to thank the project engineer David Fraguela all his support, done along the implementation phase of the Tag-Bot.

References

[1] Al-Khalifa H. S., Davis H. C.: Exploring The Value Of Folksonomies For Creating Semantic Metadata. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 3(1):13–39, March 2007.

[2] A.L.I.C.E.: Artificial Intelligence Foundation. `http://www.alicebot.org/`.

[3] Cleverbot: `http://cleverbot.com/`.

[4] FreeHal: `http://www.freehal.org/`.

[5] Hal U.: `http://zabaware.com/assistant/`.

[6] Hotho A., Jäschke R., Schmitz C., Stumme G.: Information Retrieval in Folksonomies: Search and Ranking. In *The Semantic Web: Research and Applications*, vol. 4011 of *Lecture Notes in Computer Science*, pp. 411–426. Springer Berlin / Heidelberg, 2006.

[7] Neves A. M. M., Diniz I., Barros F. A.: Natural Language Communication via AIML Plus Chatterbots. *V Symposium on Human Factors in Computers Systems (IHC 2002), pp. 329-346*, 2002.

[8] Program D. Getting Started with Program D.: `http://www.alicebot.org/resources/programd/readme.html`.

[9] Shepitsen A., Gemmell J., Mobasher B., Burke R.: Personalized Recommendation in Social Tagging Systems using Hierarchical Clustering. In *Proceedings of the*

2008 ACM Conference on Recommender Systems, RecSys '08, pp. 259–266, New York, NY, USA, 2008. ACM.

[10] Spiteri L. F.: The Use of Folksonomies in Public Library Catalogues. *The Serials Librarian*, 51(2):75–89, 2006.

[11] Szomszor M. N., Cantador I., Alani H.: Correlating User Profiles from Multiple Folksonomies. In *Proceedings of the Nineteenth ACM Conference on Hypertext and Hypermedia*, HT '08, pp. 33–42, New York, NY, USA, 2008. ACM.

[12] Vander Wal T.: Folksonomy Coinage and Definition. `http://www.vanderwal.net/folksonomy.html`.

[13] Veres C.: The Language of Folksonomies: What Tags Reveal About User Classification. In *Natural Language Processing and Information Systems*, vol. 3999 of *Lecture Notes in Computer Science*, pp. 58–69. Springer Berlin / Heidelberg, 2006.

[14] Vivant C.: `http://www.chipvivant.com/`.

[15] Wallace R. S.: The Elements of AIML Style. *ALICE A.I. Foundation Inc.*, 2003.

[16] Zhang G., Qu Z., Guo A.: Using Folksonomies to Improve Peer-to-Peer Knowledge Sharing. In *Information Processing (ISIP), 2008 International Symposiums on*, pp. 280–285, May 2008.

Affiliations

Fernando A. Mikic
Department of Telematics Engineering, University of Vigo, Campus Lagoas Marcosende s/n, 36310, Vigo (Spain), `mikic@det.uvigo.es`

Juan C. Burguillo
Department of Telematics Engineering, University of Vigo, Campus Lagoas Marcosende s/n, 36310, Vigo (Spain), `J.C.Burguillo@uvigo.es`

Ana Peleteiro
Department of Telematics Engineering, University of Vigo, Campus Lagoas Marcosende s/n, 36310, Vigo (Spain), `apeleteiro@gti.uvigo.es`

Marta Rey-López
Consellera de Educación e O.U., Edificio Administrativo San Caetano, Santiago de Compostela, `marta.rey@edu.xunta.es`

Received: 30.01.2012
Revised: 26.03.2012
Accepted: 23.04.2012