NPCs and Chatterbots with Personality and Emotional Response

Dana Vrajitoru
Intelligent Systems Laboratory, Computer and Information Sciences
Indiana University South Bend
danav@cs.iusb.edu

***Abstract*— Chatterbots are computer programs that simulate intelligent conversation. They are situated between games and toys, as their aim is mostly to be entertaining, but the user doesn't have to follow precise rules when playing with the program. Currently business and educational applications have started to emerge as a further development of the idea of intelligent dialog. For the game industry, they come close to the concept of NPC, or Non-Player Character, and they may become part of making such virtual beings more believable and life-like in the future. In this paper we present application introducing an emotional component designed to enhance the realism of the conversation.**

Keywords: Intelligent NPCs

I. Introduction

NPCs or non-player characters are an important aspect of games, especially in the role playing category. Their functionality is to add to the game content by providing access to the backstory, assigning and rewarding quests, and generally, offering information about the game to the player. Their conversation skills are relatively limited and are in general scripted, and context-based.

There are several foreseeable developments for NPCs, and one of them consists in expanding their dialog capabilities. It is likely that in the future, NPCs will merge with some of the functionality currently present in chatterbots.

Chatterbots are computer programs that simulate intelligent conversation. The typical execution involves an input from the user in natural language to which the program provides an answer that should be a reasonable and possibly intelligent response to the original sentence. The process is repeated while the human keeps the conversation going.

The very first chatterbot, named Eliza [1], simulated a Rogerian psychotherapist. The idea was simple and consisted in a pattern matching algorithm and sentence reconstruction following templates, with no in-depth knowledge or processing of the natural language. The program proved to be amazingly efficient in sustaining people's attention during the conversation and the success of the original program has influenced the development of many others.

Using similar ideas, Colby from the Stanford AI Lab developed *Parry*, the paranoid, in 1971. Parry is the opposite of Eliza as it simulates a patient and has been intended as a study of the nature of paranoia and is capable of expressing beliefs, fears, and anxieties [2], [3]. Among the famous chatterbots we can mention Racter, a story-teller, [4] by W. Chamberlain and T. Etter, the stated author of the book *The Policeman's Beard Was Half Constructed*. However, the authenticity of the book has been questioned since then [5].

Another chatterbot worth mentioning is A.L.I.C.E, Artificial Linguistic Internet Computer Entity (www.alice.org), that has its own development language called AIML (artificial intelligence markup language) and earned the Loebner Prize, based on the Turing Test [6], in 2000 and 2001.

Among recent developments are the virtual agents that can provide online help and customer service by incorporating knowledge about the company (www.egain.com). Some of the recent research also focuses on a definition of configurable personality for virtual characters [7], [8]. Some studies have also been conducted on the behavior of the human participants in the interaction with the chatterbots [9], [10].

The goal of the chatterbots we implemented is to simulate particular personalities, either fictional or real, mostly taken from literature, film, or television shows. This mostly applies to NPCs in adventure games inspired from an external story source, like the Lord of the Rings. We typically start from a database of sentences that can be attributed to the personality to be simulated, as for example, the text of the book or their lines from a script. The first prototypes can be found online (http://www.cs.iusb.edu/˜danav/chatterbots/). An integrated 3D environment for the chatterbot is currently under development.

The current chatterbot model represents an extension of [11]. In our primary model, we construct an answer to the human's input by a probabilistic choice between pattern matching and templates, sentence keyword retrieval based on automatic indexing, and database matching based on a personality-specific database. Several of the chatterbot construction operations have been automated, but a large human contribution is still necessary. The newest aspect of our program is represented by the emotional component designed to enhance the credibility of the character.

The emotional response in an essential component of any believable character [12], [13], [14]. The importance of this aspect has been recognized in the artificial intelligence community and several studies focused on it [15]. Among the possible applications of emotional agents and virtual characters we can cite teaching and tutoring [16], [17].

The paper is structured the following way. The second section presents the outline of the chatterbot program. The third section discusses the general chatterbot techniques we implemented. The fourth section introduces the personality database and emotional component of the program. A following section presents some experimental results and compares them with out previous work. The paper ends with conclusions.

II. The Virtual Character

The chatterbot algorithm consists in a loop reading an input from the user and generating an answer, until the user ends the dialog by either closing the browser or typing in a synonym of "bye". The program will attempt to generate an answer with a certain probability using the following methods in this order:

1) personal features database, 90% probability,
2) pattern matching and templates, 90% probability,
3) first word question-matching, using a different set of answers for inputs starting with "where" than for those starting with "how" and so on, 80% probability,
4) keyword-matching in the database created by automatic indexing, 90% probability.
5) random answer distinguishing between declarative sentences and questions, 100% probability.

The probabilities expressed in the list above are conditional. Thus, the pattern matching probability of 90% is conditioned by the 10% probability that the personal feature database will not be used, and by the event that this database did not contain a valid answer to the user's input. As the last method always succeeds, a hopefully valid answer will be returned in any case.

III. General Purpose Chatterbot Techniques

In this section we briefly introduce some of the techniques used by chatterbots, which are pattern-matching, indexing, and randomly matched answers. We classify these as general purpose techniques because most chatterbots are using a combination of them, but for the purposes of creating a personality for the chatterbot they are insufficient.

A. Pattern Matching and Templates

The pattern matching technique consists in finding one or several patterns that match the sentence entered by the user. A pattern is generally defined as a sentence in natural language in which certain parts have been replaced by wild cards that can be matched by any group of words in a matching sentence.

For each pattern defined in the database, a corresponding template is utilized to generate the answer to the sentence. The parts of the original sentence that are identified with the wild cards are first subjected to a person transformation in which words like "I, my, mine" become "you, your, yours" and the other way around.

For example, a pattern in the original Eliza program can be expressed as

*I want **

in which the '*' character can be replaced by any sequence of words. The corresponding template to generate the answer can be expressed by

*What would it mean to you if you got **

in which the '*' is replaced by the sequence of words that was matched to the wild card in the pattern.

An application of this pattern could be the following dialog:

User: I want to know how your program works.

Eliza: What would it mean to you if you got to know how my program works?

Beside the list of patterns, the original Eliza program also contained a list of sentences that can be given in answer to sentences that cannot be matched to any of the patterns, like:

What does that suggest to you ?

Please go on.

For a virtual character, the patterns are built from the database of character lines from the book or from the script. They are based on the lines of any other character that precedes the character we are developing in the dialog and the response templates are generated from the character's answer in the original dialog.

Here are some examples of answers generated with the pattern-template model. This method is still one of the best options because it uses part of the sentence provided by the user and thus the answer seems to have a strong connection to it.

Input: "*can you proceed without clearance?*"

Answer: "*we don't need clearance. we need the 16-digit code.*"

We developed an automatic pattern-template generating algorithm for this application that takes as input two sentences, the first one belonging to any character in the original script, and the second one belonging to the character that the chatterbot emulates and representing an answer to the first. Let q and a be the two sentences. The algorithm starts by identifying a sequence of substrings of q such that each of them is also a substring of a, but not necessarily in the same order, as shown in Equation 2. The sequence may not be the longest and its selection process is randomized. The algorithm avoids selecting common substrings that are composed of only words that are too common, like "the".

$$q = q_0\ s_0\ q_1\ s_1\ \dots\ q_{n-1}\ s_{n-1}\ q_n \tag{1}$$

$$\text{such that } \forall i = 0, n-1, s_i \text{ is a substring of } a \tag{2}$$

The program then generates a pattern by replacing each s_i in q by a "*", the wild card symbol that can be matched by any substring, even empty. The corresponding template is generated by replacing every occurrence of s_i in the sentence a with a symbol representing the substring index in the pattern, in our case denoted by $*\#i*$.

The algorithm is not yet sufficient to automatically generate the entire database of the chatterbot with no human intervention. After the patterns were automatically generated, it was necessary for a human indexer to verify their quality and eliminate some of them. Even so, this represents a significant improvement to the task of generating a chatterbot. Without it, the human indexer must define all of the pattern-template couples by hand. This process usually involves reading a substantial amount of text looking for pieces of dialog that can be used. The algorithm shortened the development time for the chatterbot considerably.

B. *Automatic Indexing for Chatterbots*

In the classical IR approach [18], we are given a collection of documents (ASCII text in natural language) and a query expressed by a human in natural language. The task of the system is to find the documents in the collection that are the closest match to the given query.

We extended this model to the chatterbot application by considering that each document consists of one or two sentences associated with the character in the original script, usually one of the character's lines. The user's input to the program can be seen as the query. In our case we want to find one particular document (sentence) that can be seen as a good answer to the query.

Thus, in the first step, we process all sentences available in the database by eliminating the words that are too common, such as "a", "is", "for", and removing the unnecessary suffixes and prefixes to retain only the root of each word. For example, "program", "programs" and "programming" will all be indexed under "program". We used the Porter stemming algorithm for this step [18].

In the next step, we build an inverted index database, where we store a reference to all the sentences in the database that contain any given keyword. Sometimes words that appear in the text before the sentence being processed can also be significant to it, so our indexing also takes into account the context of a document. Our model follows a Boolean indexing [19] meaning that we record only the presence or absence of a term in an indexed sentence, with no assumption as to its importance.

In contrast to the IR approaches where the goal is to provide the user with a reasonable number of documents presented in an order corresponding to their relevance to the query, in our case the program must return a single coherent answer to the user's input. The retrieval process starts by applying the same transformation to the input to extract the keywords. Based on these, we make a probabilistic choice of one of the stored sentences that are indexed by any of the terms present in the user's input. Our database is small enough that the program does not require a more refined retrieval system.

This approach presents certain limitations in its use for chatterbots. The first one is that the sentences it generates come from a static database so even if the answer is probabilistic, the limit of choice may cause repetition and decrease the life-like impression of the program. Another limitation is that having some keywords in common with the input is often not sufficient for a good answer. For these reasons, this method is given a lower priority than others. It is still useful sometimes when the user is looking for very specific type of information.

Here is an example of a good answer generated by indexing:

Input: "*Where is Michael?*"

Answer: "*nikita and michael just called. they've got the antibiotic and they're on their way in.*"

This next example can be described as accidentally good. It was indexed by the word "proceed".

Input: "*can you proceed without clearance?*"

Answer: "*no. insufficient personnel. how do we proceed?*"

Here is another example that resulted from a match of two keywords but that doesn't sound like it had much to do with the input:

Input: "*I need an address for the conference.*"

Answer: "*david zacharoff's attached to the canadian trade commission. he was in brussels the night rudy said he saw him addressing a conference. there's about 5,000 witnesses.*"

Sometimes this gives the impression that the chatterbot is either trying to change the subject, or that the answers are somewhat "off the wall". This method seems to be working better when the discussion focuses on subjects related to the character's backstory.

In the following example, the answer resulted from an indexing method that took into consideration the preceding sentence that the character was answering to. The result is that the answer seems to have nothing to do with the input:

Input: "*I need advise about changing my major.*"

Answer: "*that's us.*"

Of all the methods used by the chatterbot, the automatic indexing seems to generate the least coherent answers. This doesn't mean that the method should be completely abandoned, but rather limited to questions related to the game backstory, identifiable for example by keywords specific to certain quests, like character names and quest items.

C. *Randomly Generated Answers*

While writing a chatterbot, the programmer must take into account that there are many situations in which the sentence entered by the user can be matched neither by a pattern, nor by any of the indexed keywords. The database usually contains a number of answers that can be provided in such cases. For the original Eliza, they would be things like "*This is interesting, please continue*" and "*Can you elaborate on this?*".

We refined this technique to distinguish first between input expressed as a question or as a declarative sentence. The questions are further classified by the word they start with. Thus, a question starting with "where" can result in an answer containing a random location, while a question starting with "who" can result in the identification or description of a random person. There are also a number of answers that can be provided to questions that do not start with such special keywords, as for example, "*I don't know – yet.*"

Here are some examples of answers generated this way:

Input: "*Where is Michael?*"

Answer: "*20 degrees to the left.*"

Input: "*I'm bored.*"

Answer: "*sad.*"

Input: "*what are the latest news?*"

Answer: "*a vapor trail. whatever it is, it's getting closer. could be a nato plane on maneuvers?*"

D. Short-Term Memory

A program generating answers to the player's input procedurally is likely to generate the same answer for the same question. This undermines the credibility of the chatterbot or NPC as a live character. It is thus necessary to implement a failsafe that prevents answer repetition, even if the player keeps asking the same question.

In our latest model, the chatterbot keeps track of up to 5 of the answers it provides. The program stores a number associated in the indexing with the answers and not the actual text. This way we can prevent the chatterbot from using the same pattern twice in a row, even if it is to generate different sentences. Also, it prevents a sentence that was retrieved from the keyword indexing from being returned after a pattern has been used that was generated from the same original sentence.

IV. Chatterbot Personality

In this section we present the two components of the chatterbot personality, which are the database of personal preferences and the emotional response.

Creating a character with personality involves several components. In general, when it comes to an NPC with a three dimensional body and with a face that can be seen in detail in the program, these aspects are part of the personality, mainly the facial expressions and body movement. The character's reactions are even more important, as well as its level of friendliness, expressiveness, and the amount of dialog provided during the communication. The emotional aspect is also critical to a believable character and this feature increased the realism of our chatterbot.

A big part of creating characters with specific personality in our case if the fact that the database used for the dialog is created from an original script or book in which that character exists and has a distinguishable personality. One question that can be asked is if the templates created from the character's original dialog will generate sentences that are still consistent with the character. From the results presented in the next chapter, this seems to be indeed the case.

Some studies related to authorship [20], [21] propose statistical measures aimed to classify documents, establish authorship, or compare two documents. Such features include frequency of words or expressions, word ordering, use of conjunctions, modality, comments, and so on, and can be based on the syntax and on the semantics. Similar methods can be used in the future to determine the consistency of the chatterbot's answers with the original personality it intends to emulate.

A. Personality-Specific Database

We added a new component to the chatterbot presented in [11] which consists in a database of personal preferences specific to the personality represented by the chatterbot.

This database contains information ranging from the eating and drinking preferences, to family relations and friends. For example, our chatterbot Birky (www.cs.iusb.edu/˜danav/chatterbots/ebirkoff) likes to eat gummy bears and Oreo crackers, he has a brother named Jason, and a friend called Walter.

The program is then able to detect substrings in the user's input like ”are you” and ”is your”, which could be an indication that a question was asked about a personal preference, as for example “What do you want to eat?”. The program identifies a keyword in the sentence that indicates the type of preference being asked for. In the example it would be the word “eat”. A small database of synonyms is then used to match “eat” with “food” which is the database keyword of relevance to the question. The last step consists in retrieving a random answer from all the entries stored for this keyword in the database, as for example, “gummy bears”.

The database is constructed based on the personal inference of the author on what constitutes appropriate descriptions of the chatterbot's preferences. The implementation of this component can be extended in the future such that the personal preferences are automatically extracted from the original text if the chatterbot is constructed from an existing character from literature. This constitutes a direction for future research.

The personality-specific database also contains information about the character's occupation and hobbies, about all of the persons that this character considers as friends or not, and precise information like age and location. Some of the question in the Loebner Prize of the past year have addressed such issues. As part of our evaluation was based on these questions, this part of the database generated many of the answered that were evaluated as “good”.

As a direction of future development of this part of the program, we would like to incorporate in an NPC knowledge about other characters he might know from the same framework, story, or game. More generally, a similar algorithm can be used to retrieve specific information about the world of the game or about current events, if that is considered relevant to the NPC.

V. Emotional Component of the Chatterbot

This part of the project focuses on integrating an emotional component in the chatterbot program to partially match the program's answers and enhance the user's experience of the dialog.

This component of the program was generated in three steps: organizing variations of the chatterbot's avatar, generating and organizing a list of moods that could apply to the chatterbot, and attaching an emotional description to some of the sentences in the chatterbot database.

We organized the tree components, avatars, moods, and emotional descriptions, in five basic categories described by the set $\{fear, anger, sadness, happiness, other\}$. The fifth category includes everything that cannot be described by one of the four emotions. These categories were inspired from [22], where the emotions are identified by facial expressions and are classified in six categories, $\{surprise, fear, anger, sadness, disgust, happiness\}$.

While the original classification is more complete, we did not find any images of the character Birky showing disgust, and the distinction between the images showing fear and surprise was not clear enough to create separate categories.

We selected a number of expressive images of the character emulated by the chatterbot and organized them in the five categories described above. The images in each class except for the fifth one were then sorted by intensity.

For the second step we generated a list of about 100 different moods collected from mood descriptors commonly used in online communication like blogs, message boards, emoticons, and synonyms of the four basic categories. About a fifth of the moods couldn't be classified as any of the four basic emotions and constitute the fifth category. The moods in each class were then also sorted by intensity. For example, the "fear" class contains moods ranging from "uncomfortable" and "confused" to "shaken" or "terrified". Many of these moods reflect a combination of fear and surprise in various degrees.

In the last step we identified the sentences in the chatterbot's indexed database for which one of the four emotion categories could apply. These sentences are identified by a number and are used by the patterns, by the keyword-based indexing, and to generate random answers.

The mood-generation process for any answer provided by the chatterbot consists in three steps. First, we identify one of the five categories that applies to the answer, either as one of the four emotions if such an emotion could be identified for the sentence, or the category "other" if not. Second, a random avatar is selected from the identified category. And last, the avatar's index in the set of pictures is projected onto the range of moods for the same category, and a mood is selected within a range of 20% around the projected index. This way we approximately match the mood with the avatar's expression without providing the same avatar every time for any particular mood.

The selected avatar image is displayed, and a caption above the picture identifies the mood of the chatterbot. The program can be seen online at www.cs.iusb.edu/˜danav/chatterbots/ebirkoff/.

VI. Experimental Results

In this section we present some experimental results of our chatterbot.

A. Previous Work

We compare the current chatterbot with and without the emotional component with the previous work we presented in [11]. The chatterbots discussed in that paper can be seen online at www.cs.iusb.edu/˜danav/chatterbots/.

In the former state, our chatterbots were using pattern matching and templates that were entirely constructed by hand. The random answers used a distinction between declarative inputs and question-type inputs, with no further classification of the questions. From the various discussions conducted with the chatterbots, the refinement of this random component seems to represent an improvement to the quality of the answers.

The first chatterbots were using a very limited version of keyword indexing. We expanded this component by implementing an automatic indexing process. This component expanded the diversity of the answers, but also proved to provide the least quality in the answers. We foresee little use of this technique for the future, and the need for a much more selective indexing process.

The previous paper presented an evolutionary algorithm that allowed us to generate new sentences based on the ones retrieved from the database and enhance the diversity of response of the chatterbot. We did not include this component in the latest chatterbot, but it is a possible direction for future research. However, the automatic pattern generation and indexing, as well as the personal preference database seem to be better methods for expanding the space of possible answers for the chatterbot.

B. Chatterbot Evaluation

To evaluate the new chatterbot program with and without the emotional component, an experiment was conducted consisting in a dialog with the chatterbot with and without the emotional component. The discussion consisted of 50 inputs and answers for each version of the chatterbot. One subject participated in the experiments. A different set of input sentences was used with each version of the chatterbot, following the thread of the discussion. The answers were evaluated based on the following categories: reasonable answers, good answers, and off topic answers, that seem to have little or nothing to do with the input sentence. The judgments of the chatterbot answers were made by the subject of the experiment. Additionally, sentences that were syntactically incorrect were also counted, as well as the answers that were consistent with the character simulated by the chatterbot. Similar categories were used in [11] to evaluate the answers.

Table I shows the results of this experiment and compares the percentages in each categories with those reported in [11]. We computed the average from the previous paper of the percentages in each category, with and without the contribution of the evolutionary algorithm which is denoted by EA in the last column. From this table we remark a substantial improvement of the quality of responses provided by the latest chatterbot. The difference between the versions of the program with and without the emotional component is very small. This is due to the fact that the mood is added to the response after the response is generated, and the answer generation algorithm is identical in the two cases.

The results in last column, denoted by LB, were based on the Loebner Prize (http://www.loebner.net/Prizef/loebner-prize.html). We selected 150 questions that the referees asked the chatterbots and the humans in 2005. Birky's answers to these questions were then judged based on the same criteria as the other experiments. These results show a lower percentage of reasonable and good answers because they were not asked during a sustained conversation, nor were any of them specific to this particular chatterbot.

TABLE I

EVALUATION OF THE CHATTERBOTS WITH AND WITHOUT THE EMOTIONAL COMPONENT

	plain	emotional	previous	EA	LB
Reasonable	32%	30%	40%	42.5%	31.3%
Good	48%	50%	18%	15.5%	32.7%
Off topic	20%	20%	42%	42%	32%
In character	90%	84%	75%	79%	92%
Syntactically incorrect	2%	4%	4.5%	6.5%	4%

These experiments also provided an intuitive idea of what the emotional component adds to the program. In general, the chatterbot's expressed mood enhances the dialog experience and the impression of realism of the chatterbot. In several cases the mood also added to the significance of the answer. A few answers that would have been classified as "reasonable" without the mood were classified as "good" based on this additional information.

For example, when asked "would you like to talk about gail?", who is supposedly a female character in the story that the character had a romantic connection to in the past, Birky answered "i don't know." This answer can make sense without being especially to the point in the absence of information about the mood, which qualifies it as "reasonable". The mood expressed in this case was "rejected" which added a new significance to the answer and qualified it as "good".

The correspondence between the avatars and the mood expressed by the chatterbot was appropriate without being too repetitive. In most cases, the expressed mood was reasonable for the answer provided by the program.

VII. CONCLUSION

In this paper we presented a chatterbot application with an emotional component and personality-specific database. We are currently developing a game integrating this chatterbot in a 3D environment as an intelligent NPC.

In the paper we first introduced the decision making process to generate an answer. We followed with the basic techniques used by the chatterbot, pattern matching, automatic indexing, and random sentence matching, as well as the short term memory. The next section presented the personality database describing personal features like food preferences and family links, and the emotional component that associates a mood and a corresponding avatar to every answer returned by the chatterbot.

The results presented in the previous section are encouraging. The quality of response of the chatterbot has improved as compared to the previous models we implemented, as well as the space of possible answers for the chatterbot. Moreover, the emotional component adds another dimension of realism to the program and enhances the dialog experience. The capacity of emotional response is thus an important aspect in creating believable virtual characters.

REFERENCES

[1] J. Weizenbaum, "Eliza - a computer program for the study of natural language communication between man and machine," *Communications of the ACM*, vol. 1, no. 9, 1966.

[2] B. Raphael, *The Thinking Computer*. New York: Freeman, 1976.

[3] G. Güzeldere and S. Franchi, "Dialogues with colorful personalities of early AI," *Stanford Humanities Review*, vol. 4, no. 2, 1995.

[4] W. Chamberlain, *The Policeman's Beard is Half Constructed*. Warner Books, 1984.

[5] J. Barger, ""The Policeman's Beard" was largely prefab!" *The Journal of Computer Game Design*, vol. 6, 1993.

[6] A. Turing, "Computing machinery and intelligence," *Mind*, vol. 59, no. 236, pp. 433–460, 1950.

[7] F. Barthelemy, B. Dosquet, S. Gries, and X. Magnant, "Believable synthetic characters in a virtual emarket," in *IASTED Artificial Intelligence and Applications*, Innsbruck, Austria, 2004.

[8] A. Galvo, F. Barros, A. Neves, and G. Ramalho, "Persona-AIML: An architecture for developing chatterbots with personality," in *Proceeding of Autonomous Agents and Multi Agent Systems*, Columbia University, NY, USA, 2004.

[9] L. Saarine, *Chatterbots: Crash Test Dummies of Communication*. Master Thesis, University of Arts and Design Helsinki UIAH, 2001.

[10] A. De Angeli, G. I. Johnson, and L. Coventry, "The unfriendly user: Exploring social reactions to chatterbots," in *Proc. Int. Conf. Affective Human Factor Design*, M. G. Helander, H. M. Kalid, and T. M. Po, Eds. Asean Academic Press, 2001, pp. 467–474. [Online]. Available: citeseer.ist.psu.edu/557029.html

[11] D. Vrajitoru and J. Ratkiewicz, "Evolutionary sentence combination for chatterbots," in *The IASTED International Conference on Artificial Intelligence and Applications (AIA 2004)*. Innsbruck, Austria: ACTA Press, February 16-18 2004, pp. 287–292.

[12] J. Bates, "The role of emotion in believable agents," *Communications of the ACM*, vol. 37, no. 7, pp. 122–125, 1994.

[13] S. Brave and C. Nass, "Emotion in human-computer interaction," in *The human-computer interaction handbook: fundamentals, evolving technologies and emerging applications*. Lawrence Erlbaum Associates, Inc, 2002, pp. 81–96.

[14] N. Magnenat-Thalmann, "Creating a smart virtual personality," *Lecture Notes in Computer Science*, vol. 2773, no. 2, pp. 15 – 16, 1993.

[15] E. Oliveira and L. Sarmento, "Emotional valence-based mechanisms and agent personality," in *Lecture Notes on Artificial Intelligence*, ser. 2507, G. Bittencourt and G. Ramalho, Eds. Springer, 2002, pp. 152–162.

[16] C. Okonkwo and J. Vassileva, "Affective pedagogical agents and user persuasion," in *Proceedings of the 9th International Conference on Human- Computer Interaction*, C. Stephanidis, Ed., New Orleans, 2001, pp. 397–401.

[17] N. Person, A. Graesser, R. Kreuz, V. Pomeroy, and TRG, "Simulating human tutor dialog moves in AutoTutor," *International Journal of Artificial Intelligence in Education*, vol. 12, pp. 23–39, 2001.

[18] G. Salton, Ed., *The SMART Retrieval System - Experiments in Automatic Document Processing*. Englewood Cliffs (NJ): Prentice-Hall, 1971.

[19] G. Salton, E. Fox, and U. Wu, "Extended Boolean information retrieval," *Communications of the ACM*, vol. 26, no. 12, pp. 1022–1036, 1983.

[20] H. v. Halteren, "Linguistic profiling for authorship recognition and verification," in *Proceedings of the 42th Meeting of the Association for COmputational Linguistics (ACL'04)*, Barcelona, Spain, 2004, pp. 199–206.

[21] C. Whitelaw and J. Patrick, "Selecting systemic features for text classification," in *Australasian Language Technology Workshop*, Macquarie University, NSW Australia, 2004, pp. 93–100.

[22] P. Eckman and V. F. Wallace, *Unmasking the Face: A Guide to Recognizing Emotions from Facial Clues*. Englewood Cliffs N.J.: Prentice-Hall, 1975.