Integrating Natural Language Understanding and a Cognitive Approach to Textual Emotion Recognition for Generating Human-Like Responses

Shaikh Mostafa Al Masum, Mitsuru Ishizuka
Department of Information and Communication Engineering, University of Tokyo, Japan
mostafa_masum@computer.org, ishizuka@i.u-tokyo.ac.jp

Abstract

Till date no computer program has passed the Turing test although even simple conversational programs, so-called "Chatbots", could make people believe that they are talking to another human being. The main objective of this paper is to identify some shortcomings of existing conversational agents (e.g. Chatbots) and describe our approach to model human-like conversational agent that overcomes those limitations to some extent. Our primary focus is to sense affective information from input sentence(s) by applying a cognitive theory of emotions known as the OCC model and generate both pro-active and reactive responses according to the input. We thus aim at developing an emotionally intelligent computer program that not only "understands" what affective information is conveyed in textual messages, but also may provide automatic empathic response.

Keywords: Conversational Program, Human Computer Interaction, Natural Language Understanding, Affective Chat, Emotional Machine, Virtual Human, Artificial Intelligence.

I. INTRODUCTION

The Turing test (as described by Alan Turing in [1]), is a test to assess the aptitude of conversational capability of a conversational programs. In the test a human judge engages in a natural language conversation with two other parties, one human and the other a machine; if the judge cannot reliably tell which is which, then the machine is said to pass the test. Simple conversational programs such as ELIZA [2] were designed to be a psychotherapist and did little except echoing back comments to the human chatter. Since then several conversational programs have been developed, e.g. [3][4][5]. The Loebner prize [6] is an annual prize for the computer system that, in the judges' opinions, demonstrates the "most human-like" conversational behavior. Chatbot ALICE [7] is the winner of the Loebner prize for several times and it is based on AIML (for detail see [7]) script encoded prodigious knowledgebase containing 30,000 to 40,000 canned phrases and sentences together with a rule-based search engine.

However, in our opinion it is arguable whether ALICE understands natural language and replies adequately. Apart from ALICE there are many other Chatbots that also received prizes in the Loebner contest by proving themselves to produce human-like responses with respect to input text. But still the ultimate goal of the Turing test has not been achieved yet. We believe that all existing conversational programs lack some essential features, for example, the first limitation is natural language understanding (NLU); then lack of emotional and social intelligence; and finally natural language generation (NLG) conforming to context and content. In our opinion, since the existing programs are deficient in the above artifacts, the responses generated by the current programs are often irrelevant, awkward, repetitive and perplexing to make a human inter-actor uninterested after a while.

If machines would have a true understanding of input sentences, they would reply in a more natural way. For instance, if someone says *"Necessity is the mother of invention"*, ELIZA might respond with *"Tell me more about your family"*, based on its pattern for the word "mother" and ALICE may reply "*What else is the mother of invention?*"

Secondly, current Chatbots have very little or hardly any emotional and social intelligence to respond both affectively as well as in a social manner. For example if someone tells ALICE, *"This semester I really had a poor grade."*, ALICE replies with *"What does 'this' refer to?"*, which is neither affective nor in conformity with social etiquette. Hence studying the relationship between natural language and affective information as well as assessing the underpinned affective qualities of natural language might be a worthwhile attempt for improving interaction with users. According to a linguistic survey done by Pennebaker [8], across all of the studies described by him, people usually use less (i.e. about 4% of the written words) emotional words (e.g. adjectives), even though they may express affective contents significantly. This indicates that affective lexicons might not be sufficient to recognize affective information from texts and raises the suspicion that a machine learning [9][10] or keyword spotting [11][12] method might not perform well for this objective. In our approach we target recognition and expression of emotion by considering (1) a cognitively based approach for sensing machine's emotion from text modality and (2) a corpus of common-sense knowledge known as the Concept-Net [13].

Finally, Natural Language Generation (NLG) conforming to context and content is the task of generating natural language (e.g. text) based on machine representations of commonsense and real-world knowledge, extracted features from the input text and some rules to decide and select an appropriate response. The present conversational programs are not efficient to deal with context as well as content. For example, ALICE replies like "*What are your goals in life?*" or *"What is that?"* or *"Who told you that?"* etc., for the input "*The lecture of the professor is very hard to understand*" which is not contextually acceptable in our opinion. Although if someone proceeds further saying *"My friend attended his lecture last semester.*", the response "*How well do you know this person?*" spawns to another context and finally the conversation gets distracted both from context and content. Hence a sophisticated NLG system is needed to handle context and generate content accordingly. Context switching should be allowed by incorporating sophisticated rules and colorful word choice for certain turns in the dialogues. Like Plantec [14] we also admit that unexpected quirky identifying characteristics help user to see virtual people as human. In general an NLG needs a planner and methods to merge information in order to enable the generation of text that appears natural and non-repetitive. However, current programs cannot efficiently perform this task dynamically. Hence we consider *Content Determination* by maintaining a frame structure for each input sentence; *Sentence Aggregation* by utilizing the parsed information stored in the frame; and *Lexicalization* by inserting words to the concepts obtained from Concept-Net.

II. OUR APPROACH

In order to overcome of the aforementioned limitations we have implemented a noble architecture (Figure 1) that performs deep parsing to understand the input sentence; senses affective information to perform emotion synthesis and handles context to generate relevant responses by maintaining a template based information manipulator. Previous approaches for analyzing ('sensing') affect or social intelligence in texts have commonly employed keyword spotting, lexical affinity, statistical methods, pre-processed models, a dictionary of affective concept and lexicon, or a commonsense knowledgebase, but none of those methods considered the cognitive structure of individual emotions or their appraisal structure to assess the attitudinal quality of the text. Hence we employ the OCC emotion model [15] that considers emotions as valanced reactions to consequences of events, actions of agents and different aspects of objects.

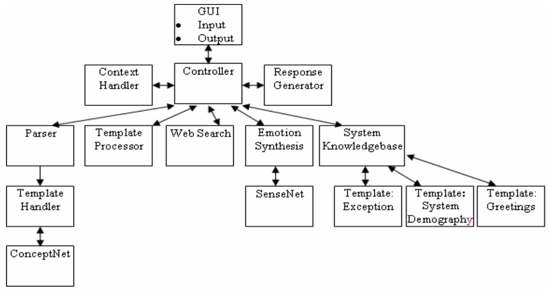

Figure 1 the architecture of the system

A. ConceptNet

ConceptNet [13] is a semantic network of commonsense knowledge that at present contains 1.6 million edges connecting more than 300000 nodes. Nodes are semi-structured English fragments, interrelated by an ontology of twenty semantic relations encompassing the spatial, physical, social, temporal, and psychological aspects of everyday life. ConceptNet is generated automatically from the 700000 sentences of the Open Mind Common Sense Project which is a World Wide Web based collaboration with over 14000 authors. It extends WordNet's [16] list of semantic relations to a handful repertoire of twenty semantic relations including, for example, EffectOf (causality), SubeventOf (event hierarchy), CapableOf (agent's ability), PropertyOf, LocationOf, and MotivationOf (affect). We employed two functions of ConceptNet, namely, DisplayNode() and GetContext(). DisplayNode() returns all the possible semantic relationships of an input concept and the GetContext() function is useful for semantic query expansion to obtain the contextual intersection of multiple concepts. As an example, for the concept "dream" we get the following concepts and contexts from the ConceptNet. Table 1 enlists few such entries for space limitation.

Table 1 Associated concept and context for "Dream"

Concept	Context
==IsA==> hope	go to bed
==IsA==> experience	go to sleep
==IsA==> personal experience	sleep
==SubeventOf==> sleep	sleep at night
==SubeventOf==> twitch	snore
==SubeventOf==> eye movement	

B. Semantic Language Parser

We are using the *Machinese Syntax* [17] program in a client-server setup, where each line or paragraph of text is sent and XML-formatted shallow-parsed information is received for further processing by our implemented deep parser written in python. For example, for the input sentence, *"I had a bad dream last night"*, we obtain XML-like syntactical information from the shallow parser, which is further processed to output as a tuple of Subject, Subject Type, Subject Attributes; Action, Action Status, Action Attribute; Object, Object Type and Object Attribute, as indicated in Figure 2.

```
[[['Subject Name:', 'i', 'Subject Type:', 'Self', 'Subject
  Attrib:', [null]]
  ['Action Name:', 'have', 'Action Status:', 'Past ', 'Action
  Attrib:', ['time: last night']]
  ['Object Name:', 'dream', 'Object Type:', 'N NOM',
```

Figure 2 sample output of semantic parser

C. Template Handler

Being inspired by the concept of frames [18], a frame-based template is instantiated for each statement that contains a different concept. Template handler assigns values to different variables of frame components from the syntactic output of the parser. The primary goal of this frame structure is to capture linguistic information along with associated concepts and context in order to record or find the answers about *"Who" "do/does" "What"*, *"Where"*, *"How" "Why"* and to/for *"Which/ Whom"*. The template shown in Table 2 with the values assigned for the above example indicates how a context is captured and the system keep itself aware of it.

Table 2 Template Structure to capture information

Frame Component	Variables to set

Event / Action	_eventName := *"have"*
	_eventPolarity := 3.947
	_eventStatus := "past"
	_eventSelf := "true"
	_isProspective := 4.474
	_isPraiseworthy: =4.211
	_attribute := *null*

Agent / Who	_experiencer : = *"self"*
	_attribute := *null*
	_agentPolarity := 5.00

Concept / What	_topic := "*dream*"
	_attribute := "*bad*"
	_topicPolarity := -3.566
	_topicDesirability:=*null*
	_topicLiking:= *null*

Location / Where	_location := *null*
	_attribute:= *null*

Temporal / When	_eventTime := *"last night"*
	_presentTime:= <sys-time>

Reason / Why	_circumstance := *null*
Target / Indirect Object	_target := *null*

Expression Type	_type := *statement*
	_pattern:= *affirmative*

Associated Concepts	[list obtained from ConceptNet]
Associated Contexts	[list obtained from ConceptNet]
Emotion Affinity	_sentimentType := [will be assigned by SenseNet] _sentimentValue:= [will be assigned by SenseNet]

The value of some of the variables are assigned by SenseNet, for example, value of _topicPolarity is set (-3.566) by SenseNet.

D. Template Processor

Template processor does mainly two functions, attempts to assign the null-valued fields of the current frame by asking the 'controller' to generate proactive response by the 'response generator' and also maintain a list of templates in order to keep track of the flow of the topic. As an example, the current template for the input *"Last night I had a bad dream"*, template processor finds that "where" field is *null* and moreover from the ConceptNet it knows that the first Sub-Event of the topic *"dream"* is *"sleep"*. So it asks the controller to generate a response by the 'response generator' using the keywords: "where" and "sleep". A question is asked by the system and the answer is then processed to be stored in the "where" field of the current template. Next the template processor may ask the controller to find about the answer to assign the value for "Why" field and hence a question like *"Why did you have bad dream?"* may be generated by the response generator and the answer is processed further. If the answer given by the person cannot be parsed successfully or unknown, template handler sends some keywords to generate responses that are contextually related. In this case if the response give by human is *"I don't know"*, response generator receives the keywords like "Bad Dream", "Desirous Effect Of", "Stress", "have wish", "Unsolved Problem" "boredom", "sexual frustration", "sleep", "intense period of work", "buy lottery ticket" etc. from this module. The keywords are obtained by applying a filtering algorithm on the list of associated concepts returned by ConceptNet for the input concept (in this case "dream" is the concept).

Template handler also manages to invoke Emotion Synthesis module to assign values for emotion affinity field when most of the fields of the template are set. For our example the system cannot decide whether the user actually desires for and like the topic. Hence, to obtain more information about the user's cognitive state, the system proactively asks by question related to the user's desirability or liking of the topic. Depending on the answers given by the user, the system then can make affective classification of the context and affective responses (e.g. sorry for or happy for etc.) can be generated.

E. Web Search

This module invokes internet search to find definition of the topic using Google's search string for finding definition. For example, http://www.google.com/search?num=10&hl=en & q=define:"+ *topic*; returns the definition about *topic*. If it fails to find any definition for a given *Topic*, ST_i, it forms sub-topics by taking the portion of the search topic and tries to retrieve definition. Algorithm to find a definition using Google is given below:

```
Begin
   Search-Topic, $ST = ST_i$
   $W_i$ is the list of words in $ST$
```

```
Search-Key, SK=NULL
Set j = C ; where is the number of words in ST
Set Definition, d=NULL
While (d=NULL)
    Search-key, $SK_j = \sum_{i=1}^{j} W_i$ ; $1 \le j \le C$
    d = getDefinitionFromGoogleFor($SK_j$)
    If (d = NULL)
        then j= j-1
    Else exit the loop
  Loop While
End
```

The motivation of this module is to find the answer about a topic when someone asks question explicitly about a topic. For example, if someone asks *"What is life?"* this module is invoked and collects maximum of five definitions and randomly selects one to answer like, *"the experience of being alive; the course of human events and activities;"*

F. Emotion Synthesis

For emotion synthesis we are following the OCC emotion model [15]. The motivation for choosing the OCC model is that it defines emotions as valanced reaction to events, agents or objects and considers valanced reactions necessary to differentiate between non-emotions and emotions. Moreover, the model constitutes a goal-, standard- and attitude-oriented emotion appraisal structure. The OCC model defines twenty-two emotion types specified by a corresponding set of lexical tokens. Due to space limitations we are providing the characterization for only one emotion type, for example, the sufficient condition for characterizing the *"Fear Confirmed"* emotion type, *Fear_Confirmed* (*a, x, e, txt*) is positive if there is an event *e*, in the text *txt*, and there is a valanced reaction found towards the event *e*, and the Experiencer *x*, doesn't desire for the event and the event described in the text has already happened and the program *a*, generally believes that the event *e*, is not beneficiary. We have implemented rules for 22 emotion types following the OCC model using several variables as mentioned in [19]. Few rules are given below.

"Happy-For" is true if senseDegree > 2.5, IsEvent = true, EventStatus='Past' or 'Present Continuous', AgentType = 'Person', likingOfEvent> = 1.0, desirabilityForOther >= 1.0 and deservingnessOfEvent >= 1.0

"Pride" is true if senseDegree >2.5, isAction=true, AgentType='Self', abs (isPraiseworthy)>=1.5

"Satisfaction" is true if senseDegree>2.5, isEvent = True, AgentType = 'Self', isProspective >= 1.5, likelihoodOfEvent >= 1, effortForEvent>=1, effortRealization >=1

For the input, "*Last night I had a bad dream"*, the program will detect a "confirmed fear" emotion if further responses of the user for the assessment of the desirability and belief factor are being assessed negatively and the system may respond to the affective state of the user, for example, by saying: *"I know, bad dream makes people afraid. Don't be scared."*

G. SenseNet

In a linguistic context, as e.g. in WordNet [16], a word sense is a given meaning of a word based on the context. Unlike WordNet, by the term "sense" used in SenseNet, we mean a lexical tuple, formed by 'a subject or agent', 'a verb or action', 'an object or concept' and associated 'adjectives or attributes' and each sense is assigned a value that we call sense-valence. SenseNet employs two lexical resources namely, WordNet and ConceptNet [13].The main idea of SenseNet [20] is to form a network of senses from the input sentence(s); to assign numerical value to each lexical unit based on their lexical sense affinity; to assess the value of the sense(s), and to output sense-valence for each lexical-unit (e.g. sentence, paragraph, and document), for detail see [20]. This module assesses the polarity of the sentence. It basically considers the <verb (event), object (concept)> pair to assess the sense considering the attributes (adjectives) and the polarity of agent as well, for example,

- Negative Verb + Positive Concept→ Negative Polarity (e.g. quit job)
- Negative Verb + Negative Concept→ Positive Polarity (e.g. quit drug, quit smoking)
- Positive Verb + Positive Concept→ Positive Polarity (e.g. buy car, save money)
- Positive Verb + Negative Concept → Negative Polarity (e.g. buy gun, encourage terrorist)

In our running example, "have" and "dream" result in a positive sense, but due to negative value of _attribute (*i.e. bad*), the combined sense becomes negative. SenseNet maintains a list of scored verbs, adjectives, concepts and named-entities. These lists are utilized with the rules to score the sentiment valence of each sentence.

H. System's Knowledgebase

The system has implemented three categories of knowledgebase namely, System demography; Greetings and Exception. These are stored in templates with question patterns and possible answers, similar to ALICE or ELIZA. System demography contains about the personal information about the program itself. From the question patterns which people usually ask a computer, we have observed that people are significantly interested to know about personal information or particular opinion from the program itself and this conforms to [21] that people most naturally interact with their computers in a social and affectively meaningful way, just like with other people. Hence to tackle the questions like "*Who are you?"*, *"Where do you live?"* etc., System demography is consulted for the answers. 'Greeting template' is instantiated to handle social greetings and to collect the demographic information of the interacting person by questioning like *"What is your favorite sport?"*, or *"What is your favorite soccer team?"* etc. based on the context of the conversation. Is the system cannot understand the input or failed to match proper context to produce response, the "Exception" template is consulted to produce responses in the cases of exception. The present idea of handling exception is implemented to reflect verbal idiosyncrasies in terms of apology or funny statements. For example, if someone inputs gibberish, the system may response with *"I am Sorry; I'm not familiar with your alien language."*

I. Context Handler

Content Handler keeps track of the current context and global context. It notifies the controller about current context to decide which module to invoke to generate response. Context handler also raises an exception if context is abruptly changed. For example if someone was talking about "bad dream" and suddenly says *"I am planning to buy a new car.";* the system notifies that the new context "buy, new car" has no semantic connection with the present context and hence the exception module is asked to produce response like *"You were talking about a bad dream, should we forget the matter?"* etc. Context handler implements several rules to allow context switching. One such rule is, when 80% fields of a template representation a context is set or been queried, the system will proactively ask the user to change the context. For space limitation all the rules are not discussed in this scope. When a new context is allowed, a new template is instantiated for the input sentence and template processor stores the previous template(s) in the list for future reference

J. Response Generator

The response generator can construct a question or a statement based on the input. In receives a pattern of keywords from controller and then produce a sentence. For example, if controller gives the following pattern, *[Agent: I, Query: where, Key: sleep, Tense: past, Time: last night]*, the response generator can generate a question like *"Where did you sleep last night?";* similarly for the pattern, *[define: love, text: have a great affection or liking for; "I love French food"; "She loves her boss and works hard for him"],* it possibly can generate response like, *"I would say love as to have a great affection or liking for; for example, "I love French food" or "She loves her boss and works hard for him"*. The response for exception is generated with the help of the exception module. For example if someone says *"Which is bigger, my hand or yours?"*, the parser says the controller that it is a question about a comparison and controller notifies it as an exception and look for the rule to handle such exception in the exception module. Finally pattern, *[exception: question, type: compare, object1: my hand, object2: yours, rule: logical size]*, is given to the response generator to handle the exception and hence a response like *"Tell the size of your hand and mine to answer logically"* may be generated. Similarly for the input *"Necessity is the mother of invention"*; may generate a reply like: *"I agree that mother of invention is conceptually related with necessity. Can you give an example?"*

K. Controller

The controller acts like the navigator of data inside the system. It receives the input sentence from the GUI and gets it parsed by the parser. Then it decides whether the input is a question or a statement. If input is a question, it further checks if the question is about the system itself or the person himself or a specific topic or some other logical question. According to the specific question type specific modules are been queried to produce keys or rules which are given to the response generator to generate reply for given input. If the input is a statement, the controller decides whether the input describes a fact associated with the person himself or the system itself or greeting or something/ someone else. Based on the decision, a context is set and a contextual template is instantiated. The template processor with the help of the context handler then produces keys that are given to the response generator to produce context relevant reply.

B. GUI

At present our Graphical User Interface (GUI) is simply a text editor where one can type a line of text and on pressing the return key the system considers that an input is given and the input is processed for an appropriate response. In future we plan to incorporate a 3D character agent with the support of Text To Speech (TTS) engine to articulate the response generated by the system in order to make the application more appealing an interactive.

III. CONCLUSION

We are considering sensing the empathic information from the text and/or making empathic reply by programs (e.g. Chatbot) to improve interaction and make text based interface like instant messaging system, *yahoo messenger* for example, more robust and interactive by associating emoticons or animating own-avatar with respect to sensed emotion to express the affective state. We enlist below a machine conversation (partial) having empathic ingredients. The conversation is made with ALICE Chatbot which is based on AIML (Artificial Intelligence Markup Language) script.

Conversation 1:

Human: *Ok, I am fine now but last night I had a bad dream*
Computer: *Glad to hear it, Human. Why now? Interesting. Really. Do you still have a bad dream?*
Human: *not really, last night I was falling from a mountain in the dream*

Computer: *Oh really? Really. Where were you falling from a mountain in the dream? I have been missing you*
Human: *that was not real but it was really scary though*
Computer: *Oh Give me an example of real but it was really scary.*

Our target is to generate reply something like the following based on sensing the empathy and other personality processes described in [8]

Human: *Cool!! But last night I had a bad dream.*
Computer: *Ahh! Was that scary?*
Human: *Not really*
Computer: *Then, why are you afraid of?*
Human: *I was in a kind of afraid of falling from a mountain!!*

Computer: *umm, Do you have such bad dream often?*
Human: *Not that often*

In our opinion, machine will be able to generate empathic reply based on our sentence level emotion sensing approach. For the above scenario, "fear-confirmed" emotion type is sensed for the input "*Cool!! But last night I had a bad dream*". So instead of replying from AIML based pre-scripted answers the machine tries to ask questions about the affective state of the user. As we are planning to implement an online education counselor (text based) for the students to discuss in a chatting manner, we think this sort of affective state awareness and machine reply will improve the interaction and usability of the system. If the input text expresses a particular affective state (for example, if "*happy-for*" emotion is detected, machine will express "*happy-for*" emotion, whereas for *"anger"* machine will express neutral affect), the system responds accordingly by querying about the affective state of the user. For example, in this case, machine asks, "*Ahh! Was that scary?"* instead of replying from ALICE- knowledgebase. The system is currently being implemented and we hope to report on its results soon.

REFERENCES

[1] A. Turing, 1950. 'Computing Machinery and Intelligence', *Mind* 59(236): 433–460.

[2] J. Weizenbaum, 1966. ELIZA -- A computer program for the study of natural language communication between man and machine, *Communications of the ACM* 9(1):36-45.

[3] A. D. Angeli, G. I. Johnson, and L. Coventry, 2001. The unfriendly user: exploring social reactions to chatterbots, In *Proceedings of The International Conference on Affective Human Factors Design,* Asean Academic Press, London.

[4] T. W. Bickmore, 1999. Social Intelligence in Conversational Computer Agents, *Gesture & Narrative Language Group*, MIT Media Laboratory.

[5] http://www.jabberwacky.com/

[6] Loebner Prize Home Page, http://www.loebner.net/Prizef/loebner-prize.html

[7] ALICE official web site, http://www.alice.org

[8] J.W. Pennebaker, M.E. Francis, and R.J. Booth, 2001. Linguistic Inquiry and Word Count: LIWC 2001. Mahwah, NJ: Erlbaum Publishers.

[9] F. Sebastiani, 2002. Machine Learning in Automated Text Categorization, *ACM Computing Surveys*, 34(1):1-47, March.

[10] P. Jacobs, 1992. Joining statistics with NLP for text categorization, In *Proceedings of the Third Conference on Applied Natural Language Processing*, Morristown, NJ, USA, pp. 178-185.

[11] A. Valitutti, C. Strapparava, and O. Stock, 2004. Developing Affective Lexical Resources, *PsychNology Journal*, 2(1): 61-83

[12] A. Boucouvalas and X. Zhe, 2002. Text-to-emotion engine for real time internet communication. In *Proceedings of International Symposium on CSNDSP,* pp. 164-168, Staffordshire University, UK, July.

[13] H. Liu and P. Singh, 2004. ConceptNet: A Practical Commonsense Reasoning Toolkit, *BT Technology Journal*, 22(4):211-226, October. Kluwer Academic Publishers.

[14] P. M. Plantec, R. Kurzwell and R. Kurzweil, 2004. *Virtual Humans: A Build-It-Yourself Kit, Complete With Software and Step-By-Step Instructions*. AMACOM, American Management Association, New York, USA.

[15] A. Ortony, G. L. Clore, and A. Collins, 1988. *The Cognitive Structure of Emotions*, Cambridge University Press, Cambridge, UK

[16] C. Fellbaum C., (Ed.). 1999. *WordNet: An Electronic Lexical Databases*, MIT Press, Cambridge, Massachusetts.

[17] Connexor Oy 2006. web site, http://www.connexor.com

[18] FrameNet, http://framenet.icsi.berkeley.edu/

[19] M. A. M. Shaikh, H. Prendinger and M. Ishizuka. A Cognitively Based Approach to Affect Sensing from Text, In *Proceedings of 10th Int'l Conf. on Intelligent User Interface*, pages 349-351, Sydney, Australia, February 2006, ACM.

[20] M. A. M. Shaikh, H. Prendinger and M. Ishizuka. SenseNet: A Linguistic Tool to Visualize Numerical Valence Based Sentiment of Textual Data. Submitted for ICON 2006.

[21] B. Reeves and C. Nass, 1998. *The Media Equation. How People Treat Computers, Television and New Media Like Real People and Places* (CSLI Publications, Center for the Study of Language and Information. Cambridge University Press.