A Very Modular Humor Enabled Chat-Bot for Japanese

Jonas Sjöbergh
Hokkaido University
js@media.eng.hokudai.ac.jp

Kenji Araki
Hokkaido University
araki@media.eng.hokudai.ac.jp

Abstract

We present a chat-bot for Japanese that is very modular. Modules for specific inputs can easily be integrated, e.g. to test methods in context without dealing with unrelated inputs. To make the chat-bot more entertaining we have several humor modules: a database of jokes; creation of jokes based on user input; and recognition of jokes from the user. These modules leave the interaction to the general modules when the context is not appropriate for their humor results. Our system outperforms two other chat-bots for Japanese, one of them also using humor, in human evaluations.

1 Introduction

Computational humor is a somewhat neglected research area that has not seen that much effort or generated that many important results. There has however been some work done, see (Binsted et al., 2006) for a good overview of the field.

A lot of research has been done on task oriented dialogue systems of various kinds, but less work has been done on non-task oriented systems, also called chat-bots. The most famous system is most likely ELIZA (Weizenbaum, 1966), which imitated a therapist and could talk about any topic by simply reframing the user inputs as questions or requests for more information. Another system that has also achieved impressive results is the A.L.I.C.E chat-bot [1]. Both these systems are based on rules and patterns written by hand by humans. For unrestricted domains this of course requires quite a lot of work to achieve good performance.

Since there has been quite little work done in both these fields, it is not surprising that there is also quite little work on chat-bots using humor though some work has been done. There have been studies showing that the use of humor can have a positive influence even in task oriented systems (Morkes et al., 1999) so it seems likely that humor is useful in non-task oriented dialogue too. There are a few chat-bots that use humor in various ways (Loehr, 1996; Augello et al., 2008; Dybala et al., 2008).

We have developed a chat-bot for Japanese, where there are no advanced systems available as far as we know. There are versions of ELIZA but nothing comparable to A.L.I.C.E. Our chat-bot is very modular and can thus easily be extended with new methods, adapted to new tasks, or personalized. Unlike the best performing chat-bots like A.L.I.C.E., the modules currently in our system are not based on manual work. There are some modules that use manually coded rules, but these are very simplistic and very little effort has been put into them. The main work is done by modules using statistics and large corpora. While manual work will probably give better results than the statistical modules we currently use, for unrestricted domains it is hard to find enough resources to achieve good performance on new languages.

2 The Chat-Bot

The chat-bot is made up of a framework for driving the conversation and different modules for handling different types of user inputs. The general framework just reads user inputs and forwards them to the modules. These then return appropriate replies and estimates of how confident they are that their reply is relevant. The confidence measures are constrained to lie between 0 and 1, so as to be comparable between the different modules. Each module is also informed if it was the module selected to reply to the last input or not.

The modules are also weighted, so when several modules have the same confidence that they have

[1] Richard Wallace, The Anatomy of A.L.I.C.E. http://www.alicebot.org/anatomy.html

a relevant reply to give, the framework selects the most favored module. This allows for easy personalization of the chat-bot, e.g. giving low weight to the word play joke modules for users that do not enjoy such jokes but high weight to these modules for users who do like word play jokes. Currently the choice of module is based on the product of the module weight and the reported confidence from the module.

The framework makes it very easy to add new modules for new types of functionality in the chat-bot. It is for instance possible to add complete task oriented dialogue systems as modules. When the system detects that the user wants some task performed (e.g. asking for tips on restaurants) control can be given to an appropriate task oriented system and returned when the task is completed. Below we describe the different modules used in our evaluation experiments.

2.1 Greeting Module

The Greeting module simply deals with greetings such as "Hello". It has a small database of common greeting phrases in Japanese and reasonable replies to these. When a greeting phrase is input it randomly selects a reply from the list of relevant ones. In these cases it reports a confidence of 1. When the input is not recognized as a greeting phrase the module returns a confidence of 0.

2.2 Aizuchi – Backchannel Module

The *Aizuchi* module always outputs *aizuchi* or backchannels, common in Japanese. For any given input, the module randomly selects general feedbacks like "really?", "uhu". It always reports a confidence of 1, and is currently used as a fallback module. When no other module has anything relevant to say, this module will produce some general statement.

More advanced strategies could of course be implemented, for instance determining if backchannels would be appropriate based on the user input. After a direct question it may not be a good idea to reply with "uhu", for instance. There has been research done on generating *aizuchi* and when to use it, see for instance (Takeuchi et al., 2002; Kita and Ide, 2007). We plan to extend this module in the future, based on such research.

2.3 Weather Module

Based on previous research by others on similar chat systems, we have found that it is fairly common to talk about the weather with chat-bots. The Weather module spots weather related keywords and replies with general statements related to these, for example replying with "I hear it will be cloudy tomorrow." when the input contains the word "sunny". If it spots a weather keyword it reports a confidence of 1, otherwise 0.

More advanced methods could for instance use any of the many weather reporting services available on the Internet to generate true replies about tomorrows weather instead of the current randomly selected replies. As the current version of the chat-bot is intended for offline use, this is not implemented yet.

2.4 Web N-Gram Model Module

The Web N-Gram Model module extracts all the content words (nouns, verbs, adjectives) from the user utterance using the MeCab[2] morphological analyzer for Japanese. It searches a large corpus of downloaded web pages for pages with all the content words. We have downloaded 20 gigabytes of web pages in Japanese, which makes slightly over 1 million pages. These are indexed by the Hyperestraier [3] search engine, which is used to search for pages with the extracted content words. From these pages the module extracts sentences that include at least one of the content words from the user input and from these sentences it builds a word trigram model.

Given this trigram model of sentences related to the contents of the user input it generates a new sentence by starting with either one of the content words followed by the Japanese topic marker "*ha*" or by two content words following each other (normally an adjective followed by a noun or one more adjective) that have non-zero probability in the model. It then randomly adds more words to this based on the word trigram probabilities gathered, until it selects a sentence ending punctuation marker. If the sentence grows longer than 20 words without generating a sentence end the sentence is abandoned and the process is restarted. If no sentence can be generated in 10 tries, the method fails and reports a confidence of 0. Otherwise the generated sentence is returned with a confidence of 1. We plan to normalize the probabilities from the generation model and use these

[2]MeCab: Yet Another Part-of-Speech and Morphological Analyzer, http://mecab.sourceforge.jp/

[3]Hyper Estraier: a full-text search system for communities, http://hyperestraier.sourceforge.net/index.html

for the confidence of the module in the future.

Generation based on n-gram statistics often give bad results, but in the evaluations the module generated many good replies, for example: *"Sore ha umai desu ka?"* ("Was that a good joke?") *"Umai umai to omotta!"* ("Yeah, I thought it was really really good!"). Longer sentences contained more meaningless examples, so perhaps making the sentence length limit even shorter would be good.

2.5 Trivia Modules 1 and 2

The first Trivia module searches the same corpus of downloaded web pages as the Web N-Gram module, but the Trivia module searches for trivia related to the user input. This module too extracts content words (currently nouns, verbs, and adjectives) from the input and searches for occurrences of these content words and phrases like "Did you know that ... <content word>?" If content words from the user input are found in such sentences, one such sentence is randomly selected as a reply and the confidence is reported as 1. Otherwise the confidence is set to 0.

All pages containing the trivia information trigger pattern "did you know that ..." and at least one user input content word are searched with regular expressions to find actual sentences that contain trivia on the relevant content words. Any trivia sentence that has already been used in the conversation is ignored.

The second Trivia module also searches the same collection of downloaded web pages for trivia like information. The only difference is that instead of searching for sentences on the form "Did you know ...?" it searches for explicit trivia statements like "Trivia: ...!".

2.6 Database Joke Module

The Database Joke module also extracts content words from the user input. It has a database of almost 3,000 jokes in Japanese. If there are jokes containing any of the input content words, one of these jokes is selected and returned with a confidence of 1. Jokes that contain more content words from the input are preferred over jokes with fewer, other than that the reply is selected randomly. Any joke already used in the conversation is ignored. Since writing jokes too often is not very funny, the module also lowers its confidence by half if the system output a joke in response to the last user input. We use the joke database collected in (Sjöbergh and Araki, 2008), which contains word play jokes.

2.7 User Jokes Module

The User Jokes module uses a very simple method to detect if the user is joking. It simply checks if the user input is present in the joke database. If so, it returns replies like "hahaha" or "That was really funny!" with a confidence of 1. This module is intended to capture when the user responds in kind to the puns output by the system.

There has been research done on using statistical means to recognize jokes, mainly for English, (Mihalcea and Strapparava, 2005; Sjöbergh and Araki, 2007), and such work has even been used in chat-bots (Augello et al., 2008). We do not yet have an implementation for Japanese, but will add this capability later to extend the amount of jokes the system can understand.

2.8 Similar Dirty Word Joke Module

The Similar Dirty Word Joke module is very much inspired by the "proverb joke" generation used in (Sjöbergh and Araki, 2008). We have adapted the method to work on any user input, not just proverbs. Other than that the method is the same.

The module uses a dictionary of dirty words to find two or more content words in the user input that sound similar to something naughty. A simple threshold on how similar the pronunciation of two words must be, and a simple list of similarity of different sounds in Japanese, are used to determine if a dirty word sounds like an input word.

The dirty words are grouped into three categories: sex related, feces related, and insults. If two or more dirty words from the same category are found and (1) sound similar enough and (2) have the same word class as the original content words, the sentence is returned with the original words replaced by the similar sounding dirty words. The reply also contains a short comment along the lines of "Oh, at first I thought you said <changed sentence>".

There has been work done on automatic generation of jokes, see for example (Binsted, 1996; Sjöbergh and Araki, 2008). Such work has been used in chat-bots (Augello et al., 2008), though only on a superficial level (like "Tell me a joke!", "Do you want to hear a joke about layers or blonds?" "Blonds" "<joke>"). As far as we know, there are no other systems for context dependent joke generation.

2.9 User Says Strange Things Module

The User Says Strange Things module is intended to respond to strange utterances by the user by saying "Isn't that a bit strange?" or something similar. It extracts the last verb from the user input and the noun followed by the particle "*wo*" (the direct object) and the noun followed by the particle "*de*" (marking instrument or place).

It then searches the same corpus as used by the Web N-Gram and Trivia modules for occurrences of this verb with these nouns in the same functions. Since there is a very great data sparseness problem when searching for raw words like this, the module returns a confidence of 0 if there are very few hits for any of the words. If all words are common but the combination of the three words is rare the user utterance is considered suspicious and the module returns a statement indicating what part of the statement was rarest, on the form "painting with a spoon, isn't that a bit strange?".

2.10 Already Mentioned Module

The Already Mentioned module notices if the user inputs something that either the system or the user already said before. If so, it returns a reply like "You already told me that." with a confidence of 1. It ignores any input that has no content words, since inputting "yes" or "really?" several times is not something to point out.

3 Evaluation

3.1 Performance Compared to Other Systems

We evaluated the system by comparing it to other previously created chat-bots for Japanese. There are no advanced chat-bots available for Japanese as far as we know, though there are some ELIZA-like systems. The first system we compare our system to is Modalin (Higuchi et al., 2008), a chat-bot that uses Internet searches to find words related to the user input. It then constructs replies by filling out patterns with strongly related words and adding modality modifiers (like "..., I think."). Modalin has been evaluated as more interesting to chat with than Japanese versions of Eliza.

The second system we use in the comparison is Pundalin (Dybala et al., 2008). It builds on the Modalin framework and adds humor generation to the chat-bot. Pundalin has been evaluated as more interesting than Modalin.

First we evaluate how interesting our system is compared to these existing chat-bots. We show three conversations, one conversation between a user and each system, to evaluators (different people than the users who chatted with the systems) and ask them which system they think is the most interesting based on the conversation logs. We kindly received such chat logs from the creators of the other systems and asked students (the users in the previous studies where also students) to chat with our system to generate comparable chat logs.

It may seem strange to ask evaluators to evaluate transcripts of conversations with the systems instead of asking the users that chatted with the systems what they think of the systems. The reason for this is that while we can have users chat with our system, the other chat bots are not possible for us to use for various reasons, but the creators of those systems gave us chat transcripts from their own evaluations instead.

For each system we used five different conversations, and each conversation was required to be at least 10 speaker turns long (a few where 11 or 12, since some users wanted to round off the conversation in some cases). Each evaluator of a set of three conversations were asked the following questions: (1) Which conversation did you think was the most interesting? (2) If you were to have a conversation with one of these three systems which system would you prefer to chat with? (3) Do you want to continue reading the conversation with system A/B/C? [from 1, meaning no interest, to 5 meaning high interest] (4) Did you think the conversation of system A/B/C was interesting? [from 1 to 5] (5) Did you think the conversation with system A/B/C was easy to understand? [from 1 to 5] (6) Did you think system A/B/C was human like? [from 1 to 5]

20 evaluators, an even mix of men and women, all of them university students, took part in the evaluation. The results of this evaluation are shown in Table 1. For all evaluation criteria our system received the highest score. Using the Students t-test, the difference between our system and the second best system (Pundalin) is significant in all cases except the "Do you want to continue reading?" question, where the difference is not significant. Thus, the system is rated as significantly more interesting, more human like, and easier to understand.

Our system also received 60% of the votes for

Question	Modalin	Pundalin	System
Read more?	2.5 (1.1)	2.7 (1.2)	3.5 (1.3)
Interesting?	2.5 (1.3)	2.7 (1.5)	4.0 (1.1)
Understand?	1.9 (0.9)	2.1 (0.9)	3.1 (1.2)
Human like?	2.3 (1.1)	2.6 (1.0)	3.5 (1.2)
Most Inter.	3	5	12
Chat with?	3	5	12

Table 1: The average scores from 1 to 5 (and standard deviation) of the three chat-bots on four evaluation criteria, and the number of votes each system received as most interesting system or system the evaluator would most like to chat with.

the most interesting system and 60% of the votes for the system that the evaluators would most like to chat with. This is more than twice as much as the second most popular system, which was Pundalin with 25% of the votes. This shows that our system outperforms the currently available chat-bots in Japanese.

It can also be noted that using humor in the conversation seems to have a positive influence. Pundalin, that also uses humor but is otherwise almost identical to Modalin, is rated higher than Modalin for all evaluation criteria. Of course, not all people enjoy the type of word play jokes used in these chat-bots and Modalin is sometimes selected as the most interesting system. Our system is easy to adapt to such users by simply giving much lower weight to the humor related modules, or by adding modules that produce different types of jokes.

3.2 Applicability of Modules

We also checked the applicability and weighting of the modules, i.e. how often a certain module can produce output in response to the user input. Since we had fairly little data from chats with our system, we also input 260 input statements that users made to a different chat-bot in another experiment. Which module responded what number of times and the weights used is shown in Table 2.

The modules that do most of the work are the Database Joke module, the Web N-Grams module, and the first Trivia module. With the chat logs from the old experiment the other Trivia module and the Weather module are also selected quite often, as is the *Aizuchi* (backchannels) module which is the fall back module mainly used when no module can generate a good reply.

Module	Weight	Now	Logs
Greeting	10	3	2
User Jokes	9.5	0	0
Trivia 1	9	8	16
Similar Dirty Word	8	2	3
Database Joke	7	26	121
Isn't That Strange?	6	0	0
Trivia 2	6	1	10
Weather	5	0	19
Web N-Grams	4	14	78
Already Mentioned	4	0	0
Aizuchi	2	0	11

Table 2: The weights of the modules and the number of times they were selected as the best output in the current experiment and when inputting old chat logs.

The Database joke module works quite well but is over used, giving too many jokes which is unnatural. The Web N-Grams module often produces good results, especially when returning short replies. On longer sentences, word n-grams are not enough and the results are often meaningless. The Trivia modules produce interesting output but suffer from the fact that the users normally respond with "No, I did not know that, tell me more!" for which some unrelated output is generated. The modules that are rarely called generally perform well when they have the chance to do something but lack opportunities. Some things are easy to improve, for instance the User Jokes module that failed to recognize a few jokes input by users because the users used different punctuation than the jokes in the database.

The modularity of the system makes it easy to adapt to different users. For users that do not like word play jokes, lowering the weight of the Database Joke module to 3 instead of 7 gives very different output. It only responds 23 times instead of 121. The remaining times are covered mainly by the Web N-Grams module, but also modules like the Weather, *Aizuchi*, and the Already Mentioned modules become more active. In the future we would like to do this kind of tuning automatically, using emotion analysis systems to analyze the user responses and rank up modules to which the user responds with positive emotions and lower the weight of modules when the user responds with negative emotions or indifference.

4 Conclusions

The evaluation shows that our chat-bot for Japanese outperforms other chat-bots for Japanese, even those also using humor. It was rated as significantly more interesting, human like, and easy to understand. Though the difference was not significant, it was also the system with the highest rating of whether the evaluator would like to read more of the conversation with the system. The system was also selected as the system that the evaluators most wanted to chat with by more than twice as many evaluators as the second most popular system.

In the future we would like to improve the chat-bot in several ways. We would like to add more hand crafted rules since these can give very good results when applicable. Collecting statistics on the most common user inputs and writing rules for these is planned. We also plan to correct some of the minor problems discovered in the current evaluation and to work on the more challenging problems with the statistical modules such as the n-gram based module.

Since the chat-bot is already working, we also plan to use it as a platform to do experiments on things that require a context to work in, for instance joke timing and context based jokes.

References

Agnese Augello, Gaetano Saccone, Salvatore Gaglio, and Giovanni Pilato. 2008. Humorist bot: Bringing computational humour in a chat-bot system. In *CISIS'08: Proceedings of the 2008 International Conference on Complex, Intelligent and Software Intensive Systems*, pages 703–708.

Kim Binsted. 1996. *Machine Humour: An Implemented Model of Puns*. Ph.D. thesis, University of Edinburgh, Edinburgh, United Kingdom.

Kim Binsted, Benjamin Bergen, Seana Coulson, Anton Nijholt, Oliviero Stock, Carlo Strapparava, Graeme Ritchie, Ruli Manurung, Helen Pain, Annalu Waller, and Dave O'Mara. 2006. Computational humor. *IEEE Intelligent Systems*, 21(2):59–69.

Pawel Dybala, Michal Ptaszynski, Shinsuke Higuchi, Rafal Rzepka, and Kenji Araki. 2008. Humor prevails! – implementing a joke generator into a conversational system. In Wayne Wobcke and Mengjie Zhang, editors, *Proceedings of AI-2008*, volume 5360 of *Lecture Notes in Artificial Intelligence*, pages 214–225. Springer.

Shinsuke Higuchi, Rafal Rzepka, and Kenji Araki. 2008. A casual conversation system using modality and word associations retrieved from the Web. In *Proceedings of EMNLP 2008*, pages 382–390.

Sotaro Kita and Sachiko Ide. 2007. Nodding, aizuchi, and final particles in Japanese conversation. *Journal of Pragmatics*, 39(7):1242–1254.

Dan Loehr. 1996. An integration of a pun generator with a natural language robot. In *Proceedings of the International Workshop on Computational Humor*.

Rada Mihalcea and Carlo Strapparava. 2005. Making computers laugh: Investigations in automatic humor recognition. In *Proceedings of HLT/EMNLP*. Vancouver, Canada.

John Morkes, Hadyn Kernal, and Clifford Nass. 1999. Effects of humor in task-oriented human-computer interaction and computer-mediated communication: a direct test of SRCT theory. *Human-Computer Interaction*, 14(4):395–435.

Jonas Sjöbergh and Kenji Araki. 2007. Recognizing humor without recognizing meaning. In Francesco Masulli, Sushmita Mitra, and Gabriella Pasi, editors, *Proceedings of WILF 2007*, volume 4578 of *Lecture Notes in Computer Science*, pages 469–476. Springer, Camogli, Italy. URL `http://dr-hato.se/research/clip2007.pdf`.

Jonas Sjöbergh and Kenji Araki. 2008. A complete and modestly funny system for generating and performing Japanese stand-up comedy. In *Coling 2008: Companion volume: Posters and Demonstrations*, pages 109–112. Manchester, UK.

Masashi Takeuchi, Norihide Kitaoka, and Seiichi Nakagawa. 2002. Implementation and evaluation of an aizuchi generation system using prosodic information. *Information Processing Society of Japan*, 2:101–102.

Joseph 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.