Extracting *Dafare* Candidates from the Web
- Japanese Puns Generating System as a Part of Humor Processing Research

Pawel Dybala¹, Michal Ptaszynski¹, Rafał Rzepka¹, Kenji Araki¹

¹Graduate School of Information Science and Technology, Hokkaido University
Kita 14 Nishi 9, Kita-ku, 060-0048 Sapporo, Japan
{paweldybala, ptaszynski, kabura, araki}@media.eng.hokudai.ac.jp

Abstract. In this paper we introduce an algorithm for extracting *dofare* (Japanese puns) candidates using the Web and Idapic Dictionary for their phonetic selection. The research is a part of PUNDA Project, aimed to create a functional humor-equipped conversational system. At its current state, our system can be used as a *dofare* generating support tool, providing the user with an output of plausible pun candidates, which can be used to generate jokes. Here we present an upgraded version of the system, with a new technique of *dofare* generation (mora transformation) implemented.

Keywords: humor, jokes, puns, pun generators, *dofare*

1 Introduction

1.1 Humor makes things better

No quotation or reference is needed to state that humor is one of human natural ways of behavior. Most agreeably, it is also a factor that enhances relations between people. The need of humor in human-computer relations has been proved by an experiment conducted by Mortes, which showed that humor-equipped devices are of higher likeability and therefore can be evaluated higher than those without humor [1].

Human-machine communication is not a science-fiction any more. Nowadays, it has become a part of our daily life, and thus it is crucial to make this feature of our reality as comfortable as possible.

1.2 Humor for medical purposes

Although healing potential of laughter has been commonly known from centuries, only recent research showed that its role can not be neglected in modern medicine. Humor therapy proved to be successful even in treatment of such serious diseases as
cancer, not mentioning mental disorders [2]. This gives humor research projects a new, medical dimension, which, combined with an urgent need of enhancing daily human-machine communication, makes our goals even more important. As laughter was proved to have therapeutic effect, any talking machine (such as, for example, car navigator), when equipped with humor, can simply make its users feel better.

1.3 Computing Japanese linguistic humor

Among all products of humor, known as “jokes”, so-called “linguistic jokes” seem the most likely to compute, as Natural Language Processing is a well-defined field of AI science. Comparing to other languages, Japanese provides its speakers with relatively big amount of homophones (not mentioning sound similarities), which makes it is a perfect surrounding for our research. Therefore, for the subject of our research we chose Japanese puns called *dajare*.

2 Existing pun generators

Although humor processing is still a heavily neglected field of NLP, some efforts to create simple pun-generating engines (computer systems that would be able to create, not simply perform recorded puns) have been made. Most of them cover simple and context-independent jokes, classified by Ritchie as self-contained puns (in opposition to contextually-integrated puns) [4]. One of the first such systems was JAPE [3] and its later Japanese conversion BOKE [5] by Kim Binsteed and Osamu Takizawa, with covered a small category of so-called punning-riddles ("How is a nice girl like a sugary bird? -Each is a sweet chick"). Although evaluation experiment showed that both systems successfully generated utterances recognized as puns, their quality was considerably lower than of those created by human.

BOKE and JAPE, as well as latter programs, like HCPP [6] and WISCRACI [7], based on the same rule of finding related sets of items and slotting them into utterances. All of them generated simple, self-contained puns.

The device proposed by Tanizawa was a humor-equipped QA system, able to answer questions with utterances containing dajare. This system is worth mentioning as an attempt of generating puns that are slightly integrated with the context, or mini-context, formed by a simple QA dialogue. However, the funniness of these humorous answers was evaluated as rather low, with 10% of AH (Average Humor) Rate [8].

3 Punda Project

Although some of existing systems succeeded in generating puns, most of them were rather simple and of low quality. Therefore in our previous publications [9,10] we introduced PUNDA (PUN-Dajare) - a research project aimed to create Japanese pun generating module (computer system able to generate puns), and implement it into a non-task oriented conversational system. We propose some new concepts to
achieve the goal, such as complex "da\-jare types set", Internet-based lexical corpus or individualized sense of humor patterns. The innovations of our approach are described in details in [9]. The system described in this paper has also been developed as a part of PUNDA Project.

PUNDA Project includes following steps:

Step 1: Dajare Types Set Extraction – based on phonetic classification of da\-jare, proposed in our previous research. See section 4.1 and 4.2 for details.

Step 2: PUNDA Module construction – using phonetic patterns extracted in step 1, pun generating module will be constructed. This step is currently under development; see section 4 for its current results.

Step 3: Preliminary research – evaluation of human-made da\-jare, which will allow us to create some models of "linguistic sense of humor". The survey on the subject is now being conducted.

Step 4: Integration with ML-Ask system – the problem of joke timing will be solved by implementing Ptaszynski's Emotive Analysis System (currently under development) into PUNDA Module [11].

Step 5: Joking conversational system – PUNDA Module will be implemented into a non-task oriented conversational agent. As the result, it will be able to generate jokes and insert them smoothly into the conversation with the user. This, when succeeded, will make PUNDA a contextually integrated pun generator.

For the time being, Step 1 has been accomplished and Steps 2, 3 and 4 are currently under development. System described in this paper has been constructed as the part of Step 2 of the project. At its current stage, it can perform a function of da\-jare generating support tool (a tool, which provides the user with a list of pun candidates, generated using the algorithm described below – see section 4.4 for the example of candidates list). To its previous version, described in [10], we added some improvements, such as new da\-jare generation pattern (mora transformation), with an attempt of Kana-Kanji conversion (see section 4.1 and 4.2 for details).

4 Our System

4.1 Dajare Types Set extraction

Step 1 of our research is based on da\-jare phonetic classification, proposed as a result of our previous research [12]. From such sources as Internet or books about da\-jare, we gathered puns created by human and divided them into 12 groups (with internal subgroups), basing on mora changes that occur between the base word and the word used in the joke. As there is no firm definition of the genre, we empirically treat all Japanese linguistic (word-plays based on language features) jokes as da\-jare.

In previous version of the system (see [10] for details) we implemented six da\-jare generation patterns from groups i-iii of our classification: i. homophony, ii. mora addition: i.a initial mora addition, i.b final mora addition, ii.c internal mora addition, iii. mora omission: iii.a final mora omission, iii.b internal mora omission [12].
achieve the goal, such as complex “dajare types set”, Internet-based lexical corpus or individualized sense of humor patterns. The innovations of our approach are described in details in [9]. The system described in this paper has also been developed as a part of PUNDA Project.

PUNDA Project includes following steps:

Step 1: Dajare Types Set Extraction – based on phonetic classification of dajare, proposed in our previous research. See section 4.1 and 4.2 for details.

Step 2: PUNDA Module construction – using phonetic patterns extracted in step 1, pun generating module will be constructed. This step is currently under development; see section 4 for its current results.

Step 3: Preliminary research – evaluation of human-made dajare, which will allow us to create some models of “linguistic sense of humor”. The survey on the subject is now being conducted.

Step 4: Integration with ML-Ask system – the problem of joke timing will be solved by implementing Ptaszynski’s Emotive Analysis System (currently under development) into PUNDA Module [11].

Step 5: Joking conversational system – PUNDA Module will be implemented into a non-task oriented conversational agent. As the result, it will be able to generate jokes and insert them smoothly into the conversation with the user. This, when succeeded, will make PUNDA a contextually integrated pun generator.

For the time being, Step 1 has been accomplished and Steps 2, 3 and 4 are currently under development. System described in this paper has been constructed as the part of Step 2 of the project. At its current stage, it can perform a function of dajare generating support tool (a tool, which provides the user with a list of pun candidates, generated using the algorithm described below – see section 4.4 for the example of candidates list). To its previous version, described in [10], we added some improvements, such as new dajare generation pattern (mora transformation), with an attempt of Kana-Kanji conversion (see section 4.1 and 4.2 for details).

4 Our System

4.1 Dajare Types Set extraction

Step 1 of our research is based on dajare phonetic classification, proposed as a result of our previous research [12]. From such sources as Internet or books about dajare, we gathered puns created by human and divided them into 12 groups (with internal subgroups), basing on mora changes that occur between the base word and the word used in the joke. As there is no firm definition of the genre, we empirically treat all Japanese linguistic (word-plays based on language features) jokes as dajare.

In previous version of the system (see [10] for details) we implemented six dajare generation patterns from groups i-iii of our classification: i. homophony, ii. mora addition: ii.a initial mora addition, ii.b final mora addition, ii.c internal mora addition, iii. mora omission: iii.a final mora omission, iii.b internal mora omission [12].
In current version of the system, we added another pattern: iv. mora transformation. This gives us an amount of seven dajare generation patterns that are used in the next step.

4.2 Extracting phonetic candidates

In our system, patterns described in previous section are used to extract dajare phonetic candidates. By “extracting phonetic candidates” we understand generation of all phonetic pun possibilities, according to the input phrase. For example, for the word ookami (a wolf), the process goes as follows:

- base word: {ookami}
- candidates:
  1. homophony: {ookami}
  2. initial mora addition: {*ookami} (aookami, iookami, uookami...)
  3. final mora addition: {ookami*} (ookamia, ookamii, ookamiu...)
  4. internal mora addition: {o*ookami}, {oo*kami}, {ooka*mi}, {ooka*mi}
     (oookami, ooookami, ouookami...)
  5. final mora omission: {ooka}
  6. internal mora omission: {okami}, {oomi}
  7. mora transformation: [{*}ookami], {o[*]kami}, {oo[*]mi}, {ooka[*]}
     (koookami, soookami, toookami... – with [*] meaning transformed mora)

As a result, we acquire a list of generated candidates. For groups 1-6, plausibility of candidates is checked in the Internet in a “threshold test”. All phrases with Yahoo hit rate higher than 5000 (set experimentally) are extracted as possible pun candidates and pass to the next stage of the system.

For group 7, however, the threshold test does not seem to be the proper solution, because of relatively high level of creativity of puns created by mora transformation. Therefore, another way needed to be found in order to state whether the candidate is plausible for the pun or not. To achieve this goal we decided to use MeCab-skkserv Kana-Kanji Converter [13], converting words written in Hiragana into Kanji. List of possible transcriptions acquired in this step is then analyzed by POS and morphological analyzer MeCab [14]. Candidates that are not divided by MeCab (which means they are recognized as one word) are extracted as “exactly matched” plausible pun candidates. From other candidates, only these with the lowest division rates are extracted, and their coocurrence is checked in the Internet. The output of this step is a list of candidates starting from the highest coocurrence rate. Also, all word transcriptions acquired from MeCab-skkserv are checked in the Internet, and another list is created, containing the candidates sorted from the highest Yahoo hit rate. These three lists (hit rate ranking, coocurrence ranking and exact match list) form a set of plausible puns propositions for group 7.

However, candidates from groups 1-6 still need to be filtered, as many of them contain base word with a particle (for example, ookamiga : <wolf> + subject particle), which obviously pass the threshold test. An algorithm for this selection is described in section 4.4.
4.3 Evaluation

Having accomplished the previous step of our research, we needed to find out if the path we choose is the right one. To do this, we used our *dajare corpus* and selected base words for jokes from groups ii-iii (mora addition, mora omission and mora transformation). Then we used these words as an input for our system and checked, if proposed pun candidates lists contained the word actually used in the joke from the corpus. As a result, 74% of candidates were found in the system output, which gave us an accuracy promising enough to continue our research. [10]

4.4 Plausible candidates' selection

The aim of this step is to eliminate these candidates that contain base word with a particle or another part of speech, such as parts of adjectives (for example, *lookami*, where -i may come from a dictionary form of Japanese i-adjective). Obviously, such candidates can not be used to generate puns, and thus they should be deleted from the list. To do this, we implemented a "snippet test" algorithm.

This problem, however, does not concern group 7 candidates, where the technique (mora transformation) by itself excludes the identity with the base word.

For each candidate from groups 1-6, that passed the threshold test, a proper Yahoo snippet is extracted and analyzed with MeCab. If any of the lines is identical with the base word, the candidate is erased. Next, if any of the lines is identical with the candidate, it is extracted as a plausible pun candidate (we call it "exact match candidate"). Then, for all other candidates, additional nine snippets are extracted and analyzed with MeCab. If any combination of joint lines includes the candidate, the cooccurrence of these phrases is checked with Yahoo, and, when there is more than one possibility of analysis, only the one with the highest cooccurrence rate is extracted.

The output of the system is the list of plausible *dajare* candidates extracted for the input word. For example, for the base word テント (*tento* <tent>) system proposed following pun candidates:

- **base word**: テント (*tento* <tent>)
- **exact match candidates**: てんとう (*tentoo* <to collapse>), パテント (*patento* <patent>), テントリ (*tentori* <competition for score>), ステント (*stento* <stent>)
- **combined match candidates**: てんと <視点+と> (*sitento*, divided into *siten* <point of view> and particle *to* <with> or quotation), んてんと <寒天 + と> (*ntento*, divided into *kanten* <vegetable jelly> and particle *to*), てんとん <店 + とん> (*tenton*, divided into *ten* <shop> and *ton*)
- **mora transformation candidates**: ていと <帝都> (*teito* <imperial capital>), テンド (*tendo* <tend>), えんと <円と> (*ento* <yen> and particle <to>), てんこ <点呼> (*tenko* <roll-call>), てんの <店の> (*tenno* <a shop> and a possessive particle *no*)

The list of mora transformation candidates is much longer, and the choice of the right one would strongly depend on the context and semantic relations with base word and other parts of the utterance, which require some further research.
5 Conclusion and future directions

In this paper we present current state of PUNDA System. Although still under construction, the system can be used as dejare generating support tool, providing the user with the list of plausible pun candidates that can be used to generate puns.

As the development of PUNDA Project will continue, we are going to implement further dejare generation patterns from our classification. Finding the semantic connection between the base word and the candidates, and choosing the most proper one would also be quite challenging task. At current point of research we are planning to use the EDR, however, some research on this matter is needed. Another important part of the project would be combining PUNDA System with ML-Ask Emotive Analysis System, which is currently under development [11]. This will solve the dejare timing problem, basing on emotive analysis of utterances. Eventually, the final goal would be implementing PUNDA Module into non-task oriented conversational system and thus creating a hum-equipped machine.

References