Computational Creativity: Automated Pun Generation

Article *in* International Journal of Computer Applications · April 2016

DOI: 10.5120/ijca2016909467

3 authors, including:

Swati Mali
K J Somaiya College Of Engineering Somaiya Vidyavihar
12 PUBLICATIONS **31** CITATIONS

SEE PROFILE

Computational Creativity: Automated Pun Generation

Priyanshi R. Shah
K.J. Somaiya
College of Engineering
Ghatkopar, Mumbai-77
Maharashtra India

Chintan D. Thakkar
K.J. Somaiya
College of Engineering
Ghatkopar, Mumbai-77
Maharashtra India

Swati Mali
Department of
Computer Engineering
K.J. Somaiya
College of Engineering
Ghatkopar, Mumbai-77
Maharashtra India

ABSTRACT
Humor is an integral part of our day-to-day communication making it interesting and plausible. The growing demand in robotics and the constant urge for machines to pass the Turing test to eliminate the thin line difference between human and machine behavior make it essential for machines to be able to use humor in communication.

Moreover, Learning is a continuous process and very important at every stage of life. However sometimes merely reading from a book induces lassitude and lack of concentration may hamper strong foundation building.

Children suffering from Autism Spectrum Disorder (ASD) suffer from slow learning and grasping issues. English being a funny language, a particular word has multiple meanings, making it difficult for children with ASD to cognize it. Solving riddles induces fast learning and sharpness in children including those affected by ASD.

The existing systems however, are too far from being used in any practical application. This paper proposes a system that uses core ideas of JAPE to create puns for entertainment and vocabulary building purpose for children.

General Terms
Homophone: Two or more words having the same pronunciation but different meanings, origins, or spelling (e.g. new and knew) [4].

Homonym: Two or more words having the same spelling or pronunciation but different meanings and origins (e.g. pole. and pole) [5].

Rhyming words: Words that have the same ending sounds. E.g. are cat, hat, bat, mat, fat and rat [6].

Punning words: A form of word play that suggests two or more meanings, by exploiting multiple meanings of words, or of similar-sounding words, for an intended humorous or rhetorical effect [7].

Pun generator: A system that uses punning words to generate riddles/jokes with an intention of making it humorous.

Keywords
Pun generator, puns, riddles, jokes, computational creativity.

1. INTRODUCTION
The power and potential of Artificial intelligence and machine learning is being understood and this encompasses automated generation of poetry or proses, automated answering machines, Chat bots etc. humans by machines for creative productivity.

Inspired by these ideas, this research aims to create an automated pun generator that will spontaneously generate puns (riddles in question answer format) using richness of English language and concepts like synonyms, homophones to help children build their vocabulary in a fun-loving way. It could also be used by old retired people as a source of entertainment.

There has been immense research in the field of computational creativity since 1992 and the JAPE algorithm[10], the only successful prototype implementation proving humor could be used by machines too was revolutionary. However there was no practical application of the prototype created as it lacked a proper GUI and the puns generated were quite often bizarre.

This research work proposes an –'Automated Pun Generator' with attractive Graphical User interface and feedback mechanism for evaluating the quality of puns generate. The system is divided into various modules as follows:

- Pun Generator
- Static puns developed by humans
- Graphical User Interface
- Top puns module
- Rating Module.

The paper consists of following sections:

Section 2: Related word focusing on describing existing system.

Section 3: Proposed system explaining the implemented algorithm in detail

Section 4: Limitations of proposed system.

Section 5: Implementation details-examples of puns generated by system

Section 6: Future Work

Section 7: Conclusion

Section 8: Acknowledgement

Section 9: References

2. RELATED WORK
Past twenty years has witnessed development of the JAPE that builds puns and has significantly contributed to the field of Computational creativity. The JAPE program (Binsted and Ritchie, 1994, 1997; Binsted, 1996)[10] is different from the early pun generators in two main aspects:

- It uses WordNet as lexicon database rather than small human-made lexicons
- JAPE jokes were funniest and covered a wide range of subjects unlike other pun-generators

The JAPE system generated three distinct classes viz:

- word substitution
- syllable substitution
- metathesis

Some of the examples of word substitution type of jokes generated by JAPE are:

1) What do you call a murderer with fiber?
   A Cereal Killer
2) What do you call a pixel that shouts
   A Computer Scream
3) What is green and bounces
   A Spring Cabbage

The JAPE system uses a noun phrase consisting of a noun and an adjective. The adjective is used as the homophone (spring in the above figure and a corresponding adverb of the alternate meaning of the adjective is extracted. The noun is then converted into an adjective by extracting the predefined characteristics of the noun-word from the lexicon database. The adverb and adjective (of the noun) are then used as keywords to generate the pun.

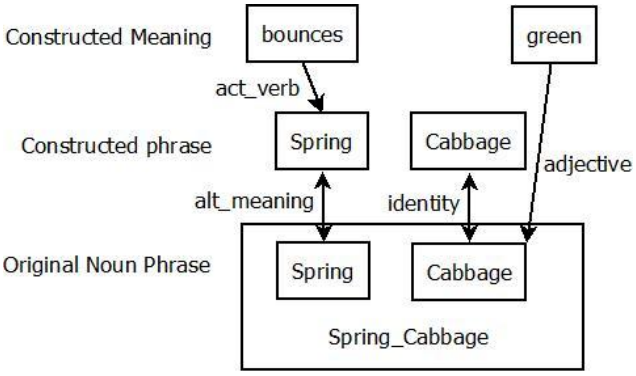

Figure 1: Example of JAPE Implementation [10].

However, the possibility of an answer using these keywords is myriad as there is no defining noun in the question. For example the above question (What is green and bounces) can also be "A tennis ball" rather than a spring cabbage. Hence a more specific representation of the noun is necessary

3. PROPOSED SYSTEM

We use the core algorithm idea of JAPE and WORDNET as the starting point. The proposed system concentrates on jokes only by word substitution as with the available resources it is easier to generate these types of puns. Moreover, a later study on JAPE demonstrated that word substitution jokes were, on an average, were funnier and more sensible than other types.

The approach follows four steps (steps 1-3 are for actual pun generation and the fourth layer is the user Interaction Layer). The actual pun generation is divided into 3 layers- word picking and keyword extraction, keywords generation, pun generation using template.

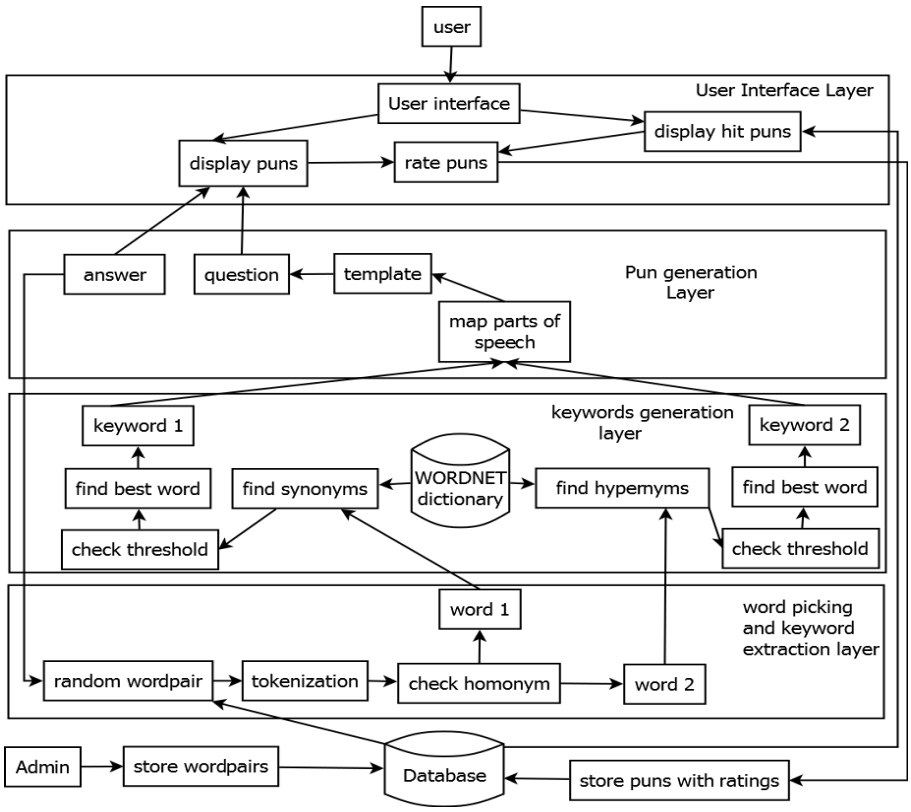

Figure 2: Architecture Diagram

3.1.Algorithm for Pun Generation

Assumption: The input is a valid multiword (e.g. "traffic pencil" is not an acceptable input for this sentence)

Input: A multiword MW1 (e.g. spring cabbage).

For the first word:

Get S = Random multiword from database

- Tokenize the words as X and Y
- Check homonym availability of X and Y and store homonymic word in X
- Hom = Get all homonyms of X
  - SHom= find all synonyms of set X
  - Hsim=For all H use WORDNET similarity metrics in Wordnet::Sim to compute target words w1
- HypY: Find all hypernyms of Y
  - Get words similar to HypY
  - Use wordnet similarity metrics in Wordnet::Sim to compute two target words w2
- Tokenize Use appropriate template with X, Y, w1 and w2 to generate output.
  - Output: "What do you get when you cross w1 and w2? Answer: MW1"

3.2.Detailed Explanation Of Algorithm

According to the architecture diagram [figure 2], the algorithm follows bottom-up approach as follows:

Layer 1- Word Extraction and Keyword Extraction:

Keyword generation is divided into 3 following stages:

1. Precondition satisfaction and tokenizing:

This includes various constraints that have to be satisfied for a particular word to instantiate keyword generation. For example the word chosen has to be a multiword (XY is a valid multi-word consisting of X and Y as valid English words individually). Another precondition is one word has to be an adjective and other should be a noun.

E.g. XY = spring cabbage

Here, multiword (spring cabbage) = true and X= spring, Y= cabbage are both valid English words

Homonym(X) = true and Noun(Y) = true

2. Homonym selection and meaning extraction (Keyword 1 generation):

The tokenized words are checked for existence of homonyms in lexical database. The lexical database is a learning knowledgebase and the corresponding contextual and alternate meaning of the searched homonym is then extracted.

Considering the above example, the precondition for Homonym (spring) is true. It's contextual meaning is 'A type of cabbage' and alternate meaning is 'season'.

3. 3)Extracting the second meaning:

As the multiword used is an adjective-noun pair, where the punning word can be either adjective or the noun (spring is an adjective that acts as the punning word in case of 'spring cabbage' and' whereas in-case of 'wrist watch' watch acts as the punning word but is a noun)

A homonym, according to the algorithm, is essentially a punning word for joke generation because it has two different root meanings for the same word. The two different meanings usually do not belong to the same part of speech. One sense takes the word as a noun and the other is usually an adjective. However there are exceptions like bat (where both are used as noun forms. The algorithm fails in such scenario) Hence to find the second meaning of the punning word (homonym extracted from the multiword) the algorithm implements parts of speech tagging (function: POStagging (String multiword, String homonym)) on the homonym and implements following algorithm.

Function def POStag(multiword, homonym):

- Use POS tagging on the multiword
- Extract the POS of homonym
- If(POS==adjective)
  - Return definition of noun form the word
    Else
  - Return synonyms of adjective form of the word

The word meaning generated is used as the first keyword for pun generation.

For example when XY= spring cabbage, X= spring Y=cabbage;

Assuming all pre-conditions are satisfied and homonymic word is identified as X, function call to POStag(spring cabbage, spring) identifies spring as adjective and cabbage as noun. Hence it returns the synonym of adjective form of spring.

POStag(spring cabbage, spring)= season

Layer 2- Keyword Generation:

The first step of the algorithm is causes the raw words (spring and cabbage in the above example) to be bound to variables X and Y. Once the second step is executed the non-homophonic word is bound to Y and is used to generate the second keyword of the pun

The algorithm finds the hypernym of the word bound to the variable Y and binds it to HypY.

The JAPE algorithm used meronyms instead of hypernyms by implementing SQL query search and checking the pre-condition meronym[Y, MerY] = true. However on a very huge lexical database the effective processing time would increase. WORDNET consists of hypernyms associated with synsets in hierarchical order making search efficient.

As metaphoric representation of a word is difficult to understand for children, using hypernyms would be more effective.

Hypernym(Y) = Hypernym(cabbage).

The above query gives result as 'cruciferous vegetable'

The extracted hypernym is used as second keyword for generation of joke.

Layer 3- Pun Generation Using Surface Templates:

A surface template is a generalized set of words containing slots within to be used for generating sentences containing various keywords. For example "what do you call a

_____ with _____" or "what do you get when you cross a _____ with a _____". The algorithm uses a set of such surface templates to generate questions for the pun. The keywords generated, along with the tokenized multiword is used to instantiate question generation. Question [spring, cabbage, season, cruciferous vegetable].

The template produces the question as "What do you get when you cross season with a cruciferous vegetable"
The restriction on template type does essentially have an effect on the quality of pun. However, using different surface templates for different types of keywords can add creativity. This makes it an ideal topic for further research.

3.3.Going Beyond JAPE

There are various limitations of the JAPE system including absence of a User interactive GUI (JAPE uses command prompt to display jokes). Also, there is no text-to speech module for the system to be used by the blind and the system is mundane. The proposed system tries to overcome these limitations is the following ways:

3.3.1. Integration of Existing Puns

There are numerous popular puns developed by humans that are never old irrespective of the era of living. For example the evergreen knock knock joke:

Knock Knock
Who's there !
Ice cream !
Ice cream who ?
Ice cream if you don't let me in!

Using these jokes will hence improve the overall performance of the system. The existing human/machine made jokes are collected from numerous sources ensuring a huge variety is available. [2][3].

3.3.2. User Interaction

The interaction between the system and user is done via a bright, colorful children-friendly GUI specially designed for children to provide sense of control to them. The jokes are broadly divided in four categories viz: knock knock, animals, sports and general. A request for a type of joke can be made using the available buttons by mouse clicks or using keyboard. Another important feature is the "top puns" category. This category displays the best jokes available that are already generated by the system and previously rated by the user. This feature improves the overall performance of the system, giving user a rich experience in a limited time span.

3.3.3. Keeping Interest in the System

Most of the systems become boring after using for few days as they lack the motivational factor. The proposed system provides a text box (not available for knock-knock jokes) asking user to answer the punning question to make it more interactive. Moreover on every correct answer, a clapping noise will be played in the background, augmenting the interest and playfulness of the system. This motivates the children to not give up and try level best.

3.3.4. Parental Mode (Block abusive words)

The prototype of system provides a feedback module for parents to flag the abusive words that may be inappropriate for children. These words will be hence stored as blacklisted words along with the common abusive word list that is manually entered by the developers

This mechanism ensures learning is not detrimental.

3.3.5. Enhanced For Blind.

According to the August 2014 estimate by World Health Organization [5], 285 million people are estimated to be visually impaired worldwide
The system contains a text-to-speech Google API that reads out the created puns making it useful for disabled.
However, the system will not contain feature to record answers and hence section 3.2.4 will not be applicable in such scenario. This however is an ideal scope for future research.

4. LIMITATIONS

The efficiency of the proposed system is majorly dependent on the multiword chosen. If the multiword selected is not a punning word and has no homonym, the algorithm fails.

Secondly, the multiword have to be chosen very carefully as the punning word (homonym) sometimes have the second meaning that is not used very frequently. Hence it becomes difficult for understanding.

Thirdly, the wup_similarity function used finding the hypernym with meaning closest to the word, does not always yield best results that fit the context of the pun. The following are some abrupt synonyms of word extracted by the system:

5. IMPLEMENTATION AND EVALUATION

The algorithm was implemented and tested multiple times using the proposed algorithm to test the efficiency of puns generated. The number of unique puns dynamically generated by the system was close to 700 and the average rating achieved based on the puns rated by users was 2.35. Table 1 displays the puns generated by the system and the corresponding average rating given by users (>10 for each pun).

Table 1: Examples of puns generated by system

SR. NO.	PUN QUESTION	PUN ANSWER	RATING
1	What do you get when you cross bedroom furniture with an error?	A Bed Bug	3
2	What do you get when you cross a season with a cruciferous vegetable?	A Spring Cabbage	3
3	What do you get when you get when you cross a strong wind and an appliance?	A blow dryer	2
4	What do you get when you are restricted for appearing for some event and visual communication?	A bar graph	1
5	What do you get when you cross alloy and an act of	A steel bowl	2

	throwing a ball?		
6	What us reservation in advance with protective covering called?	Book cover	3
7	What do you get when you cross a person with lighter complexion and commerce trade?	Fair trade	4

6. FUTURE WORK

It would be very illuminating to carry out a long term study of the use of the software by children, to obtain some idea of the effects such language play has on linguistic, communicative or social skills. Comparisons with other educational software would be interesting, as would studies with other user populations (e.g. children with autism, second-language learners).

6.1. Implementing Phonetic similarity

As mentioned in section 4, one of the major limitations of the system is failure in-case of presence of a non-homonymic word in the multiword. Using The CMU Pronouncing Dictionary a word similar to the tokenized word (either rhyming word or Homophone) could be extracted making it possible to make the word punning with an aim to use it. For example "screen" and "scream" have similar phonetic representations.

6.2. Dynamic surface template

The template used by the system is a static template as mention in section 3.3.3. Using Natural Language Processing, creating an algorithm to fit the keywords by generating a dynamic template based on the type of word, part of speech and training set of English words and sentences; will increase the efficiency of puns generated. These algorithms can be inspired from chat-bots that generate sentence based on the keyword to be included in the answer. The algorithm of T-peg appears to be extremely useful [11].

7. CONCLUSION

The phenomenal rating for some of the jokes generated by the system proves that computational creativity in the field if machines using humor is possible. It may be at an amateur level but a significant milestone is achieved. However, there is immense research and further enhancements required for computational humor to be used by robots or chat-bots. Using of learning algorithms to learn the language and use of each word as humans do seems an unexplored path in the field of computational creativity.

8. ACKNOWLEDGMENTS

Especial thanks to Ms. Swati Mali and Mr. Aaditya Joshi for their guidance and Ms. Nirmala Shinde for her valuable insights.

9. REFERENCES

[1] S. Attardo, Linguistic Theory of Humor, Mouton de Gruyter, 1994.

[2] Funny Knock Knock Jokes- website consisting of jokes online- Searches the webpage and stores it in the database, source: www.jokes4us.com, http://www.jokes4us.com/knockknockjokes/

[3] Funny one liners, Website consisting funny one liner, http://onelinefun.com

[4] Google search Engine, www.google.com, Retrieves meaning of "homophone", https://www.google.co.in/webhp?sourceid=chrome-instant&rlz=1C1CHVO_en-GBIN585IN585&ion=1&espv=2&ie=UTF-8#q=homophones%20meaning

[5] Google search Engine, www.google.com, Retrieves meaning of "homonym", https://www.google.co.in/webhp?sourceid=chrome-instant&rlz=1C1CHVO_en-GBIN585IN585&ion=1&espv=2&ie=UTF-8#q=homonyms+meaning

[6] Google search Engine, www.google.com, Retrieves meaning of" rhyming words", https://www.google.co.in/webhp?sourceid=chrome-instant&rlz=1C1CHVO_en-GBIN585IN585&ion=1&espv=2&ie=UTF-8#q=what+are+rhyming+words

[7] Google search Engine, www.google.com, Retrieves meaning of "punning word", https://www.google.co.in/webhp?sourceid=chrome-instant&rlz=1C1CHVO_en-GBIN585IN585&ion=1&espv=2&ie=UTF-8#q=what+are+punning+words

[8] World Health Organization, http://www.who.int/en/http://www.who.int/mediacentre/factsheets/fs282/en

[9] Graeme Ritchie, Ruli Manurung, Helen Pain, AnnaluWaller, Rolf Black, Dave O'Mara-A practical application of computational humour, 2007 Goldsmiths, University of London, https://www.abdn.ac.uk/ncs/documents/ijwcc07.pdf

[10] Kim Binsted and Graeme Ritchie-An implemented model of punning riddles, Department of Artificial Intelligence, University of Edinburgh, Edinburgh, Scotland EHl 1HN, AAAI-94 proceedings

[11] Bryan Anthony Hong, Ethel Ong –"Generating Punning Riddles from Examples", 2008 Second International Symposium on Universal Communication, 978-0-7695-3433-6/08, 2008 IEEE DOI 10.1109/ISUC.2008.28