Automatic Generation of Jokes in Hindi

Srishti Aggarwal
LTRC, KCIS
IIIT Hyderabad
srishti.aggarwal@research.iiit.in

Radhika Mamidi
LTRC, KCIS
IIIT Hyderabad
radhika.mamidi@iiit.ac.in

Abstract

When it comes to computational language generation systems, humour is a relatively unexplored domain, especially more so for Hindi (or rather, for most languages other than English). Most researchers agree that a joke consists of two main parts - the setup and the punchline, with humour being encoded in the incongruity between the two. In this paper, we look at *Dur se Dekha* jokes, a restricted domain of humorous three liner poetry in Hindi. We analyze their structure to understand how humour is encoded in them and formalize it. We then develop a system which is successfully able to generate a basic form of these jokes.

1 Introduction

The Oxford dictionary has defined humour as the quality of being amusing or comic, especially as expressed in literature or speech. It is an essential element of human interactions. The understanding and sense of humour varies widely across time, space and people. The same properties which make it interesting and compelling also make its structure rich and complex, making it an interesting topic of study.

Verbal humour is the most commonly used form of humour in the everyday use of natural language. Joke, a sub-class of verbal humour, is commonly considered the prototypical form of verbal humour, produced orally in conversations or published in collections (Dynel, 2009). Most researchers (Sherzer (1985), Attardo and Chabanne (1992), Attardo (1994, 2001), Suls (1972)) agree with Raskin (1985) that jokes comprise a setup and a punchline. The setup builds a narrative and induces some form of expectation, while the punchline is the final portion of the text, which violates that expectation, thereby generating humour due to the production of incongruity and its consequent resolution.

As intelligent systems advance further and find their way into increasingly more domains the need to process human natural languages increases; a challenging part of this is processing humour. Hence, in this paper, we attempt to build a system, which generates humorous texts in a restricted domain of *Dur se Dekha* type jokes, a form of humorous three liner poetry[1] in Hindi. This kind of poetry was popular in the Indian culture at one point but is almost lost now, with only a few repeated examples available online.

2 Related Work

Till date, there has been limited contribution towards construction of computational humour generation systems. One of the first attempts in generating humour was the program JAPE (Binsted and Ritchie (1997), Ritchie (2003)). JAPE used an algorithm to generate funny punning riddles, or more specifically phonologically ambiguous riddles, having noun phrase punchlines.

HAHAcronym generator (Stock and Strapparava, 2003) was another attempt at computational humour generation, whose goal was to automatically generate new versions of existing acronyms, which were ensured to be humorous using incongruity theories.

Aside from this, Taylor and Mazlack (2004), worked on humour comprehension on a restricted domain of "knock-knock" jokes. Although the heuristic based approach was able to effectively identify the word-play instances which was the first task, it faced difficulty in the knock-knock joke identification part, again stressing the chal-

[1] A three line poem is called a tercet.

Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics- Student Research Workshop, pages 69–74
Vancouver, Canada, July 30 - August 4, 2017.
https://doi.org/10.18653/v1/P17-3012

lenge in humour analysis.

More Recently, Valitutti et al. (2016), attempts to generate humorous texts by substitution of a single word in a short text, within some constraints. The results showed that taboo words made a text recognizable as humorous.

Petrović and Matthews (2013) attempted to build a fully unsupervised humour generation system which generates a specific form of jokes using big data. This is a first attempt at unsupervised joke generation which gave promising results.

While there have been some attempts to formalize humour in English and a handful of other languages, to the best of our knowledge, no such prior research work has been attempted in Hindi.

3 *Dur se Dekha*

Dur se Dekha is a focused form of poetic three liner jokes in the Hindi language. These jokes have a typical structure, with the humour lying in the incongruity of the punchline against the expectation of the listener. A typical *Dur se Dekha* joke consists of three parts. The structure of a typical *Dur se Dekha* joke can be surmised as follows:

> Part$_1$: *Dur se dekha to* $\underline{NP_1/VP_1}$ *tha,*
> Part$_2$: *Dur se dekha to* $\underline{NP_1/VP_1}$ *tha,*
> Part$_3$: *Paas jaakar dekha to* $\underline{NP_2/VP_2}$ *tha.*

Translation:

> Part$_1$: From afar I saw $\underline{NP_1/VP_1}$,
> Part$_2$: From afar I saw $\underline{NP_1/VP_1}$,
> Part$_3$: On going closer it turned out to be $\underline{NP_2/VP_2}$

A *Dur se Dekha* joke employs stylistic features like alliteration, repetition and rhyme, which have previously been associated with humour appreciation (Bucaria, 2004; Mihalcea and Strapparava, 2006).

The first part is the setup of the joke, the second is a repetition of the first. This repetition puts emphasis on the first. The beginning of the third part raises the listeners expectation, and it is what comes at the end that creates humour in this joke. That is the element of surprise, the punch line.

Alliteration exists in the first and the third word of the template. Also, the setup and the punchline normally rhyme and contradict in meaning and sentiment, which are two more indicators of humour present in these jokes. This is also precisely why these jokes lose their humour when translated into another language.

On further analysis, we classified these jokes into two main types, which are explained below with examples.

Type 1.a:

> Part$_1$: *Dur se dekha to* $\underline{NP_1}$ *tha,*
> Part$_2$: *Dur se dekha to* $\underline{NP_1}$ *tha,*
> Part$_3$: *Paas jaakar dekha to* $\underline{NP_2}$ *tha.*

An example of the basic form of this type would be:

> Joke$_1$: *Dur se dekha to Dharmendra tha, dur se dekha to Dharmendra tha, paas jaakar dekha to bandar tha.*[2]
>
> Joke$_1$(translated): From afar I saw Dharmendra, from afar I saw Dharmendra, on going closer it turned out to be a monkey.

A more complex example of the same type is:

> Joke$_2$: *Dur se dekha to Gabbar Singh ka adda tha, dur se dekha to Gabbar Singh ka adda tha, paas jaakar dekha to aapka chadda tha.*[3]
>
> Joke$_2$(translated): From afar I saw Gabbar Singh's haunt, from afar I saw Gabbar Singh's haunt, on going closer it turned out to be your underpants.

Type 1.b:

> Part$_1$: *Dur se dekha to* $\underline{VP_1}$ *tha,*
> Part$_2$: *Dur se dekha to* $\underline{VP_1}$ *tha,*
> Part$_3$: *Paas jaakar dekha to* $\underline{VP_2}$ *tha.*

Examples of jokes belonging to this category are:

> Joke$_3$: *Dur se dekha to ande ubal rahe the, dur se dekha to ande ubal rahe the, paas jaakar dekha to ganje uchal rahe the.*[4]
>
> Joke$_3$(translated): From afar I saw eggs boiling, from afar I saw eggs boiling, on going closer it turned out to be bald men jumping.

[2] http://www.jokofy.com/2595/door-se-dekha-to-funny-shayari/

[3] http://www.shayri.com/forums/showthread.php?t=27592

[4] http://www.shayri.com/forums/showthread.php?t=27592

Joke$_4$: *Dur se dekha to hasina baal bana rahi thi, dur se dekha to hasina baal bana rahi thi, paas jaakar dekha to gaay puch hila rahi thi.*[5]

Joke$_4$(translated): From afar I saw a beautiful girl grooming her hair, from afar I saw a beautiful girl grooming her hair, on going closer it turned out to be cow swinging its tail.

Type 2: The structure being the same as Type 1, but differing in the how humour is encoded in the joke.

Part$_1$: *Dur se dekkha to NP/VP$_1$ tha,*
Part$_2$: *Dur se dekha to NP/VP$_1$ tha,*
Part$_3$: *Paas jaakar dekha to NP/VP$_2$ tha.*

For example:

Joke$_5$: *Dur se dekha to kuch nahi dikha, dur se dekha to kuch nahi dikha, paas jaakar dekha to kuch tha hi nahi.*[6]

Joke$_5$(translated): From afar I could see nothing, from afar I could see nothing, on going closer it actually turned out to be nothing.

Joke$_6$(translated): Dur se dekha to baarish ho rahi thi, dur se dekha to baarish ho rahi thi, paas gaya to bheeg gaya.[7]

Joke$_6$: *From afar I saw that it was raining, from afar I saw that it was raining, on going closer I got drenched.*

The difference between Type$_1$ and Type$_2$ jokes lies in the way humour is encoded in them. Type$_1$ jokes are humorous because of how the punch line contrasts with the subject of the setup in terms of the sentiment. The incongruity between the subject (what something seems to be from far away) versus the punchline (what it actually turns out to be), induces surprise and amusement in the listener. The use of celebrity names further makes a joke funnier. On the other hand, for Type$_2$ jokes, humour lies in the unconventionality of the punchline. The listener expects a conventional punchline, something conflicting the subject, what he gets instead is an affirmation of the subject (Joke$_5$), or maybe a consequence of the subject in the real world (Joke$_6$). This is not unlike the way humour is encoded in some shaggy dog jokes, "a nonsensical joke that employs in the punchline a psychological non sequitur ... to trick the listener who expects conventional wit or humour" (Brunvand, 1963).

For the purposes of this study, we will be looking at only Type$_1$ jokes.

4 Experimental Design

We are attempting to build a system that would generate the Type$_1$ jokes mentioned in the previous section. As of now this system generates only the most basic form of Type$_{1a}$ jokes (such as Joke$_1$), but will be expanded upon to cover as many types as possible.

The Joke generation process has four steps:

Step$_1$: Template selection
Step$_2$: Setup Formation
Step$_3$: Punchline Formation
Step$_4$: Compilation

These steps are explained in the subsections below.

4.1 Template Selection

We created a collection of three templates for the two types of jokes we are working on (Type$_{1a}$ and Type$_{1b}$), along with a few minor variations in terms of auxiliary verbs, postpositions etc. These varied templates were added to naturalize the final jokes, and as a measure against joke fatigue.

4.2 Setup Formation

We manually created a lexicon of words from different semantic categories - human and non human. For humans, we compiled a list of names of popular celebrities, both male and female, from different domains - actors, politicians, players as well as popular fictional characters. For the second category in the lexicon we picked some generic adjectives, and words (mostly nouns) from the Hindi language that would have a negative sentiment when associated with a human, for example, names of animals, or vegetables.

For the form of Type$_{1a}$ jokes we have been working on, one word is picked randomly from this lexicon. This is the setup, the subject for our joke.

[5] http://desipoetry.com/door-se-dekha-series/
[6] http://desipoetry.com/door-se-dekha-series/
[7] http://www.shayri.com/forums/showthread.php?t=27592

4.3 Punchline Formation

We select a single word from the lexicon as our punchline. This selection is done following three constraints explained below:

1. Category constraint: The semantic category of the punchline should be of a different category than the subject.

2. Gender Constraint: As all non-human things are assigned a gender out of male/female, the gender of the punchline has to be the same as the gender of the subject. This is important in Hindi as it plays an important part in subject-verb agreement.

3. Form constraint: A typical characteristic of this type of jokes is that the subject of the setup and the punchline are lexically similar, giving a poetic effect to the joke. So, we first look for a rhyming word for the punchline. If a rhyming word isn't found, we use Levenshtein distance to find a phonetically similar word which is then used as the punchline.

4.4 Compilation

The template, the setup and the punchline is taken and put together to generate the resultant joke.

Given below are a couple of examples of the jokes generated by our system using the above mentioned algorithm:

> Joke$_7$: *Dur se dekha to mowgli tha, dur se dekha to mowgli that, paas jaakr dekha to chimpanzi tha.*
>
> Joke$_7$(translated): From afar I saw mowgli, from afar I saw mowgli, on going near I saw it was a chimpanzi.
>
> Joke$_8$: *Dur se dekha to mussadi tha, dur se dekha to mussadi that, paas jaakr dekha to fissadi nikla.*
>
> Joke$_8$(translated): From afar I saw Mr. Mussadi, from afar I saw Mr.Musadi, on going near he turned out to be a loser.

5 Evaluation and Results

We evaluated our system in 2 parts. We first drew a comparison between human created jokes available and the ones generated by our system. Second, we varied our constraints to see how that lead to changes in humour perception of the resultant jokes.

Experimental Conditions	**Funniness** (mean $\pm$ std.dev.)
Form + Gender + Category	2.40 ± 0.53
Gender + Category	2.11 ± 0.54
Form + Gender	1.97 ± 0.49
Form + Category	1.68 ± 0.24

Table 1: Mean Funniness values, and standard deviations of computer generated jokes as different constraints are applied.

Since, in our sample set, we had only 5 instances of the subtype of *Dur se Dekha* jokes that we are working on, we took all of those 5 and an equivalent number of jokes generated by our system and put them in a survey in a random order. We then asked 15 participants to rate how natural each joke felt to them on a scale of 1 to 5 (with 5 being completely natural and 1 being completely unnatural).

The analyses of the responses showed us that the average score of the jokes generated by our system was comparable to the average of our exemplar human made jokes. With jokes from our system having an average score of 3.16/5, our system only marginally underperforms the human jokes with an average score of 3.33/5. Another interesting observation is that of all the jokes present in the survey, the one with the highest score was one of the computer generated jokes.

In the second part of the evaluation, we want to look at how the three proposed constraints affect the resultant text. For this, we created a set of 80 jokes - 20 from the system with all three constraints, and 20 each obtained removing one variation at a time. We then conducted a survey this entire set. Each person who then took the survey got a different randomly generated subset of jokes to rate, with each joke being evaluated by 5 people. In this survey, the evaluators were asked to judge how funny they found each joke to be on a scale of 1 to 5 (with 1 being not funny at all, to 5 being hilarious). The results of this evaluation have been summarized in Table 1.

From Table 1, we see that people found our jokes only mildly funny. We believe that the simplicity, the lack of a deeper semantic meaning is the reason for this. Varying the constraints, we see that the jokes work best when they adhere to

all three constraints. We infer from the table that Gender constraint contributes the most to humor in our jokes, while Form constraint contributes the least.

We were unable to perform an inter-rater agreement analysis because each joke was rated by a different set of people. Instead, we chose to include the standard deviations for each set in our analysis.

6 Future Extensions

Our joke generator is giving encouraging results for the basic form of the jokes but it still has a long way to go to improve domain coverage, and only then will it be possible to evaluate how well the system works.

The lexicon needs to be expanded to add adjectives, adverbs and verbs so that noun phrases and verb phrases can be formed and used as the subject and the punchline. We also plan on adding associated features of nouns, in terms of their physical representation in the world, which would help add semantic significance to the results. This will require much more sophisticated algorithms. This task is especially challenging due to a lack of availability of language resources and tools for Hindi. We will need to develop phrase generators for Hindi for the task. Also, as the punchline should have the sentiment opposite to the subject line, more thought needs to put into what that means for complete phrases.

The lexicon can be updated regularly. In fact, we can make our system such that it automatically picks up trending celebrity names, adjectives and verbs from social media websites and use them as subjects for the joke. This will be instrumental in avoiding joke fatigue and would help our system keep up with the fast changing culture these days.

Also, a much more extensive evaluation should be done for the system when it is capable of generating more complex jokes. Naturalness of the jokes, as well as their funniness needs to be evaluated on a larger scale. Using crowdsourcing for such an evaluation would be a good choice to learn more about the bigger picture.

7 Summary and Conclusion

Computational linguistics is a widely explored field these days, with translation, summarization and comprehension being a few of the many areas of interest. Because of its complexity, and huge amount of variations, verbal humour has been explored only mildly in computational language processing, with only a few attempts at generating humour. This job is also made difficult due to the lack of any robust theoretical foundations to base a system on. Further, there has essentially been no work of significance in the domain for Hindi.

Our *Dur se Dekha* joke generator is a first step towards the exploration of humour in Hindi language generation. In this paper, we took a focused form of humorous tercets in Hindi - *Dur se Dekha*, and performed an analysis of its structure and humour encoding. We then created a lexicon, and came up with an algorithm to form the various elements of the joke following specific constraints. We saw that the jokes generated by our system gave decent results in terms of naturalness and humour to serve as a baseline for future work.

Finally, we discussed possible extensions for our system to make it more complete and comprehensive.

Acknowledgments

The authors would like to thank all the evaluators for their time and help in rating the jokes. We would also like to thank Kaveri Anuranjana for all the time she spent helping us put this work on paper.

References

Salvatore Attardo. 1994. *Linguistic theories of humor*, volume 1. Walter de Gruyter.

Salvatore Attardo. 2001. *Humorous texts: A semantic and pragmatic analysis*, volume 6. Walter de Gruyter.

Salvatore Attardo and Jean-Charles Chabanne. 1992. Jokes as a text type.

Kim Binsted and Graeme Ritchie. 1997. Computational rules for generating punning riddles. *HUMOR-International Journal of Humor Research* 10(1):25–76.

Jan Harold Brunvand. 1963. A classification for shaggy dog stories. *The Journal of American Folklore* 76(299):42–68.

Chiara Bucaria. 2004. Lexical and syntactic ambiguity as a source of humor: The case of newspaper headlines. *Humor* 17(3):279–310.

Marta Dynel. 2009. Beyond a joke: Types of conversational humour. *Language and Linguistics Compass* 3(5):1284–1299. https://doi.org/10.1111/j.1749-818X.2009.00152.x.

Rada Mihalcea and Carlo Strapparava. 2006. Learning to laugh (automatically): Computational models for humor recognition. *Computational Intelligence* 22(2):126–142.

Saša Petrović and David Matthews. 2013. Unsupervised joke generation from big data. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Sofia, Bulgaria, pages 228–232. http://www.aclweb.org/anthology/P13-2041.

Victor Raskin. 1985. Semantic mechanisms of humor. *Reidel, Dordrecht* .

Graeme Ritchie. 2003. The jape riddle generator: technical specification. *Institute for Communicating and Collaborative Systems* .

Joel Sherzer. 1985. Puns and jokes. *Handbook of discourse analysis* 3:213–221.

Oliviero Stock and Carlo Strapparava. 2003. Hahacronym: Humorous agents for humorous acronyms. *Humor* 16(3):297–314.

Jerry M Suls. 1972. A two-stage model for the appreciation of jokes and cartoons: An information-processing analysis. *The psychology of humor: Theoretical perspectives and empirical issues* 1:81–100.

Julia M Taylor and Lawrence J Mazlack. 2004. Computationally recognizing wordplay in jokes. In *Proceedings of the Cognitive Science Society*. volume 26.

Alessandro Valitutti, Antoine Doucet, Jukka M Toivanen, and Hannu Toivonen. 2016. Computational generation and dissection of lexical replacement humor. *Natural Language Engineering* 22(5):727–749.