Automatic generation of poetry: an overview

Article · January 2009

CITATIONS
19

READS
541

1 author:

Hugo Gonçalo Oliveira
University of Coimbra
97 PUBLICATIONS **503** CITATIONS

SEE PROFILE

Automatic generation of poetry: an overview

Hugo Gonçalo Oliveira
hroliv@dei.uc.pt
CISUC
Universidade de Coimbra
Portugal

Abstract

This paper is about the automatic generation of creative text, more precisely the automatic generation of poetry. It starts by presenting two possible categorisations for systems that aim generating poetry and then makes a brief overview on the existing attempts to this subject based on what can be found in the literature.

1. Introduction

In the last years we have seen several attempts to generate creative objects automatically, with the help of computer programs and we have started to accept the computer has an artist. The creation of visual art, the composition of musical pieces or the generation of creative text are just some of the fields that actual creative systems deal with.

The generation of natural language is a well-established and promising sub-field of artificial intelligence and computational linguistics. Its main goal is to develop computer programs that can produce text that can be understood by humans. Among the types of text generated automatically we can find, for example, biographies [9], weather forecasts [2] and also text that includes creative features. Attempts to the automatic generation of creative text include systems capable of generating story narratives [3, 8], jokes [1, 18] or poetry.

This paper is focused on the generation of the last referred text genre. The automatic generation of poetry involves several levels of language like phonetics, lexical choice, syntax and semantics and usually demands a considerable amount of input knowledge. It is a complex task and though an interesting topic for artificial intelligence research.

We start by presenting two different categorisations for the existing poetry generation systems. The first is based on the approaches and techniques used to accomplish poetry generation and the second is based on the goals of the system and on the output text. A brief overview on several systems capable of generating poetry is then provided.

2. Categorisation of poetry generation systems

Despite not requiring exaggerate precision [5], poetic texts involve regular syntactic and phonetic patterns where rhythm, metrics, rhyme and other features like alliteration or figurative language play an important role.

Manurung [12] affirms that when it comes to writing poetic text, we need to break several rules that are usually present in the production of natural language text. He refers some specific issues that need to be taken into consideration while writing poetry:

- High occurrence of interdependent linguistic phenomena that requires consideration of semantics, syntax and lexis.
- There may not be a well-defined message.
- Rich resources are needed to satisfy phonetics, syntax and semantics.
- Objective evaluation of the output text is very difficult.

This section presents two different proposed categorisations for poetry generation systems according first to the approach taken and techniques used and then to the goals the systems try to achieve. For each of the presented categories, examples of systems that embody its properties are given. Short descriptions of all the referred systems are later provided in Section 3.

2.1 Approaches and techniques

According to Gervás [7], it is (roughly) possible to group poetry generation systems based on the approaches and techniques they use:

- Template Based Poetry Generation: templates of poetry forms are filled with words that suit the defined constraints (either syntactic, rhythmic or both). After using the Poetry Creator and looking at several generated results, it is clear that the system follows this approach.
- Generate and Test Approaches: random word sequences are produced according to formal requirements, that may involve metric, other formal constraints and semantic constraints. Manurung's chart system [11], the WASP system [5] and the generate and test strategy of Tra-La-Lyrics [15] follow this approach.
- Evolutionary Approaches: poetry generation is based on evolutionary computation. POEVOLVE and Manurung's McGonnagall [12] are two systems that follow an evolutionary approach.
- Case-Based Reasoning Approaches: existing poems are retrieved, considering a target message provided by the user, and are then adapted to fit the required content. ASPERA [6] and COLIBRI [4] are examples of this approach.

2.2 Goals

For the purpose of his thesis, Manurung defines that poetic text must hold all the following three properties:

- Meaningfulness: convey a conceptual message, which is meaningful under some interpretation.
- Grammaticality: obey linguistic conventions prescribed by a given grammar and lexicon.
- Poeticness: exhibit poetic features.

He describes a taxonomy for poetry generation systems, based on the goals they try to achieve and how their output texts embody the previously referred properties:

- Word salad: systems that simply concatenate random words together, without following any grammatical rules (none of the properties are embodied).
- Template and grammar-based: words are selected from a lexicon in order to fill gaps in sentence templates (property of grammaticality).
- Form-aware: the choice of words follows a pre-defined text form, like the haiku[1] or the sonnet. This is accomplished by following metrical rules (properties of grammaticality and poeticness). The WASP [5], POEVOLVE [10], the programs by the ALAMO group, Wong and Chun's system [19] and the generative grammar strategy of Tra-la-Lyrics [15] are examples of systems belonging to this category.
- Poetry generation: the generated text embodies all the three properties of grammaticality, poeticness and meaningfulness. ASPERA [6], COLIBRI [4] and McGonnagall [12] are included in this category.

The random words strategy of Tra-La-Lyrics [15] seems to give rise to another entry in the presented taxonomy, since the metrics of the generated text suit the given rhythm and the text contains poetic features (rhyme), but the word order does not follow any grammatical rules and there is no semantics present. In other words, this strategy satisfies the property of poeticness but none of the other two.

3. Attempts to poetry generation

The automatic generation of poetry has become a research topic in the late 1990s, when the first serious works on this subject started to be discussed in scientific meetings and published in the literature. In this section several attempts to poetry generation that can be found in the literature are briefly described.

3.1 The ALAMO group

The ALAMO group has been generating poems in French for some time. Some of the programs they use are described in their website[2]. As an example, the construction of the *Rimbaudelaires* used existing Rimbaud's sonnets as a starting point. Then, words like nouns, verbs and adjectives were replaced by words belonging to Baudelaire's poetry vocabulary. The replacements are claimed to follow "strong syntactic and rhythmic constraints".

3.2 Manurung's chart system

In 1999, Hisar Manurung applied chart generation to generate natural language strings that satisfy given rhythmic patterns.

The using of charts for parsing is quite common. This technique consists of storing all complete constituents once they are found or constructed so that they can be reused, eliminating the inefficiency of backtracking. Chart generation is basically using a chart parser in the opposite direction: while a parser analyses strings and translates them to logical forms, a generator translates logical forms to strings.

[1] Form of Japanese poetry consisting of three metrical phrases of 5, 7 and 5 syllables respectively.

[2] http://indy.culture.fr/alamo/rialt/pagaccalam.html (former URL) http://alamo.mshparisnord.org/rialt/index.html (April 2009)

In this work, sentences are logically represented by first order predicates describing the input semantics. In order to deal with the rhythm, lists of 'w' (weak syllable) and 's' (strong syllable) were used to represent stress patterns. The stress patterns of the words used were based on the lexical stress and can be obtained from a pronunciation dictionary. For example, the stress for the word *incumbent* would be [w, s, w] and for the *interrupt* [w, w, s].

During the generation, when the result of a new rule is obtained, the respective stress pattern is appended to the existing one. Then, before adding the result to the chart, the new stress pattern is checked for compatibility with the target pattern. Only results with compatible patterns are added, ensuring that the generated texts satisfy the pattern.

This system was used in the further developed poetry generation system, McGonnagall.

3.3 POEVOLVE

In Levy's [10] work, a computational model of poetry generation, based on the theory of evolution, is discussed. The real process of human poetry writing is taken as a reference from which to draw the intuitions that drive the system.

The architecture of a system following Levy's model would include:

- One (or more) generator module: for creating the initial population of candidate poetic objects and modify these objects in the subsequent generations;
- Evaluator modules: for analysing and selecting the highest ranked individuals at each generation;
- A work space: where the population resides;
- A lexicon, a conceptual knowledge base and a syntactical knowledge base.

The POEVOLVE system is a prototype implementation of the architecture proposed by Levy, that creates texts that satisfy the form specifications of limericks[3].

The initial population is created from a set of words that include phonetic and stress information. Appropriately rhyming words that can appear at the end of each line are selected and then more words are selected to fill the rest of the line based on their stress information. A genetic algorithm is employed and evolution is achieved by mutation and crossover operators that modify the words contained in the limericks. Evaluation is performed by a neural network that was trained on human judgements of how creative each one of a selection of limericks is. This prototype, however, doesn't take syntax and semantics into account.

[3]Five-line poem with a strict form AABBA

3.4 Poetry Creator

The Poetry Creator is a simple system that generates poetry based on words describing a subject, a synonym for the subject and a title for the poem, all three given by the user. The resulting text consists of pre-defined verse templates where gaps are filled with the words given. Despite being referred by Manurung [14], the only information about this system was obtained by using the applet in the Poetry Creator website[4].

3.5 WASP

The WASP system [5] was one of the first serious attempts on automatic poetry generation. It is a forward reasoning rule-based system that aims to study and test the importance of the initial vocabulary, the word choice, the verse pattern selection and the construction heuristics taking into account the acceptance (or not) of the generated verses and complete poems.

The system's input consists of a set of words and a set of reference verse patterns. A given block of text is splitted into shorter fragments and all the words are collected and can be used in the poem. The obtained fragments are used to produce the reference patterns. The output of the system can either be a set of verses that satisfy the constraints of some strophic form (such as *romances*, *cuartetos* or *tercetos encadenados*) or a set of free verses.

The WASP is in fact a set of programs, each one implementing a different construction heuristics for the generation of poems. It works on the draft of the current verse and follows a generate and test approach.

The basic verse generation algorithm starts with the selection of the appropriate pattern. Then, a word corresponding to the first category of the pattern is randomly chosen from the vocabulary and appended to the draft. At each stage the draft is tested against the conditions it should satisfy. If the conditions are not met, it is rejected and a new verse starts being generated.

The WASP was used to make experimental generations in order to evaluate strategies for:

- Independent verse generation: avoiding word repetition and validation of the draft (3 different possibilities for each);
- Complete poem generation: selecting both the pattern and the rhyme for the next verse.

3.6 ASPERA

ASPERA [6] is a forward reasoning rule-based system that, given a prose description of the intended message

[4]http://www-cs-students.stanford.edu/~esincoff/poetry/jpoetry.html

and a rough specification of the type of poem, selects the appropriate metre and stanza, generates a draft poem, requests modification or validation of the draft by the user and updates its database with the information of the validated verse.

The construction strategies included are an improved version of the ones developed in WASP. Words must be combined according to the syntax of the language and should make sense according to their semantics. When occurring at the end of lines, words may have additional constraints imposed by the strophic form. In ASPERA no rich lexicon, syntax or semantics are involved, so it relies on engineering solutions to achieve equivalent results without attempting to model the complexity of natural language. The selection of words uses methods based on similarity calculations between the intended message and a base of validated verses.

The basic unit for poem composition in ASPERA is the line and not the sentence. The generation process starts by interacting with the user to obtain the specification of the poem, given by the following parameters:

- Approximate length of the poem (number o lines);
- Rhyme structure;
- Degree of formality;
- Setting (urban or rural);
- Mood (positive or negative).

The length of the poem and the degree of formality are used to search in the knowledge base for the most appropriate strophic form, while setting and mood are used in the vocabulary selection. Besides these parameters, the user is also asked to provide a prose paraphrase of the intended message that will be used in the planning of the poem draft. Each line is generated with a case-base reasoning (CBR) approach:

1. Retrieve step: for each sentence in the intended message, a specific verse is retrieved from a corpus of verse examples.
2. Reuse step: a draft of the line is constructed based on the part-of-speech (POS) structure of the chosen verse as a template.
3. Revise step: the draft is presented to be validated or corrected by the user.
4. Retain step: validated poems are analysed and stored in order to add the corresponding information to the verses database, making it possible to reuse them in further generations.

3.7 COLIBRI

COLIBRI [4] is a poetry generation system very similar to ASPERA. It also uses the CBR method for generation but the cases are stored in a very flexible representation using a Description Logic System. COLIBRI incorporates an application-dependent ontology, called CBROnto, that improves the inference power of the system as well as the representation and use of more explicit and general knowledge.

3.8 McGonnagall

Hisar Manurung's McGonnagall is the result of an evolutionary approach to generate poetry, that is described in his thesis [12]. Although the thesis is dated from 2004, a computational model for poetry generation [14] and an architecture based on the model [13] were presented earlier, in 2000.

Manurung formulated the poem generation process as a state space search problem using stochastic hill-climbing search, where a state in the search space is a possible text with all its underlying representation, and a move can occur at any level of representation, from semantics to phonetics.

The stochastic hill-climbing search model is an evolutionary algorithm with two stages: evaluation and the evolution. A set of evolutionary individuals is formed based on initial information, target semantics and target phonetics. During the evaluation, the individuals are scored based on different aspects, such as surface form, phonetic pattern and semantics. In the evolution stage, individuals are selected according to their scores. The subset consisting of the individuals with higher scores is selected for reproduction to produce mutated and, hopefully, better versions of the poem. Since mutation may occur at different levels of representation, operators must take special care to preserve consistency. The operators presented in [14] include a semantic explorer, a semantic realiser and a syntactic paraphraser.

In McGonnagall, phonetic patterns are represented in the same way they are in Manurung's chart system [11]. The grammars are represented with the Lexicalized Tree Adjoining Grammar formalism and flat-semantics is used, where each individual is basically associated with a set of propositions.

All the three properties that, according to Manurung [12], poetic text must hold (referred in Section 2.2), are satisfied by McGonnagall's output.

3.9 Tra-la-Lyrics

Tra-La-Lyrics [16] is a system that aims to generate text based on the rhythm of a song melody, given as input. After analysing the lyrics of a set of songs written in Portuguese,

it was observed that, most of the times, strong beats in the rhythm are associated with the lexical stress in the words. Using the sequence of strong and weak beats as a rhythmic pattern, the task of generating song lyrics is very similar to the generation poetry.

Taking that into consideration and adding other usual features in song lyrics like rhymes and repetition, three strategies, with different complexities, were developed in order to generate the lyrics [15]:

- Random words: the only constraints when choosing the words are rhythmic. It is however possible to setup the probability of reusing words and the probability of having rhymes in specific places.
- Generative grammar: the word choice follows not only rhythmic constraints but also syntactical constraints (given by sentence templates). The syntactical constraints take priority over the rhythmic constraints, but the rhythm is still suited most of the times. If none of the constraints can be satisfied, backtracking is used. This strategy also supports the setup of the probability of reusing words, probability of using words with given roots and it also tries to stop grammatical sentences in the same beats that musical sentences end.
- Generate and test: words following sentence templates are generated and evaluated against musical sentences. After several generations, the sentence that fits better the rhythm is chosen.

3.10 Generation using vector space model

Wong and Chun [19] present an approach to generate "modern haikus" using text found in blogs. They believe that with this approach the generated haikus will be more human-understandable and will have more realism than traditional approaches that, eventually guided by provided keywords or patterns, build poems from the scratch.

The proposed approach uses a keyword lexicon and a line repository. The keyword lexicon consists of 50 words commonly used in haiku writing while the line repository contains fragments of sentences, found in blogs. These fragments are originated from the segmentation of sentences where the keywords are used and have four words at most.

The haiku generation process starts by choosing three keywords from the lexicon to form the general picture. Then, fragments using these keywords are searched in the line repository. Two keywords are extracted from each one of the given fragment, using a weighing scheme to evaluate how important a word is in a sentence. In the final step, vectors are used to describe the semantic relationship of sentence pairs. For each possible pair of sentences, a query is created, with a keyword from each sentence. The result of this query in *Yahoo!* is assigned to the corresponding element in a vector. The sentences chosen for the resulting haiku are the ones with the most semantically related pair of vectors.

4. Concluding remarks

The automatic generation of poetry is a complex and interesting topic for research since it involves several levels of language. However, what makes this task even more interesting, is that some of those levels do not really have to be strictly present. Two different categorisations of poetry generation systems were presented in this paper and actual systems were described. As one can notice, while some approaches simply rely on satisfying the metrics and including poetic features, others are more concerned with syntax and semantics and yet they are all valuable contributions, since there are different genres of poetry and, usually, the transmitted message does not even have to be always completely accurate.

As it happens for other creative objects, it is difficult to objectively evaluate the quality of a poem. There is however work on defining objective criteria for assessing how creative the output of a system is [17]. Despite being impossible to state that computer generated poems have reached the quality of poems written by humans, there is no doubt that the whole set of attempts on this subject provided rich contributions for research on this area and were able to produce very interesting results.

References

[1] K. Binsted. *Machine humour: An implemented model of puns*. PhD thesis, University of Edinburgh, Scotland, 1996.

[2] L. Bourbeau, D. Carcagno, E. Goldberg, R. Kittredge, and A. Polguère. Bilingual generation of weather forecasts in an operations environment. In *Proceedings of the 13th conference on Computational linguistics*, pages 318–320, Morristown, NJ, USA, 1990. Association for Computational Linguistics.

[3] S. Bringsjord and D. A. Ferrucci. *Artificial Intelligence and Literary Creativity: Inside the Mind of BRUTUS, a Storytelling Machine*. Lawrence Erlbaum Associates, Hillsdale, NJ., 1999.

[4] B. Díaz-Agudo, P. Gervás, and P. A. González-Calero. Poetry generation in colibri. In *ECCBR '02: Proceedings of the 6th European Conference on Advances in Case-Based Reasoning*, pages 73–102, London, UK, 2002. Springer-Verlag.

[5] P. Gervás. Wasp: Evaluation of different strategies for the automatic generation of spanish verse. In *Wiggins, G. (Ed.). Proceedings of the AISB00 Symposium on Creative & Cultural Aspects and Applications of AI & Cognitive Science, Birmingham, UK*, 2000.

[6] P. Gervás. An expert system for the composition of formal spanish poetry. *Journal of Knowledge-Based Systems*, 14:200–1, 2001.

[7] P. Gervás. Exploring quantitative evaluations of the creativity of automatic poets. In *Workshop on Creative Systems, Approaches to Creativity in Artificial Intelligence and Cognitive Science, 15th European Conference on Artificial Intelligence*, 2002.

[8] P. Gervás, B. Lönneker-Rodman, J. C. Meister, and F. Peinado. Narrative models: Narratology meets artificial intelligence. In *Basili, Roberto and Lenci, Alessandro (Ed.). International Conference on Language Resources and Evaluation. Satellite Workshop: Toward Computational Models of Literary Analysis, Genova, Italy*, 2006.

[9] S. Kim, H. Alani, W. Hall, P. H. Lewis, D. E. Millard, N. R. Shadbolt, and M. J. Weal. Artequakt: Generating tailored biographies with automatically annotated fragments from the web. In *Semantic Authoring, Annotation and Knowledge Markup (SAAKM) 2002 Workshop at the 15th ECAI*, pages 1–6, 2002.

[10] R. P. Levy. A computational model of poetic creativity with neural network as measure of adaptive fitness. In *Proceedings of the ICCBR-01 Workshop on Creative Systems*, 2001.

[11] H. Manurung. A chart generator for rhythm patterned text. In *Proceedings of the First International Workshop on Literature in Cognition and Computer*, 1999.

[12] H. Manurung. *An evolutionary algorithm approach to poetry generation*. PhD thesis, University of Edinburgh, 2004.

[13] H. Manurung, G. Ritchie, and H. Thompson. A flexible integrated architecture for generating poetic texts, 2000.

[14] H. Manurung, G. Ritchie, and H. Thompson. Towards a computational model of poetry generation. In *AISB00 Symposium on Creative & Cultural Aspects and Applications of AI & Cognitive Science, 17th-18th April 2000, U. of Birmingham, England.*, 2000.

[15] H. R. Gonçalo Oliveira, F. A. Cardoso, and F. C. Pereira. Exploring different strategies for the automatic generation of song lyrics with tra-la-lyrics. In *Neves, J. & Santos, M. & Machado, J.M. (Ed.). New Trends in Artificial Intelligence, pp. 57-68, Guimares, Portugal*, 2007.

[16] H. R. Gonçalo Oliveira, F. A. Cardoso, and F. C. Pereira. Tra-la-lyrics: an approach to generate text based on rhythm. In *Cardoso, A. & Wiggins, G. (Ed.). Proceedings of the 4th. International Joint Workshop on Computational Creativity, London, UK*, 2007.

[17] G. Ritchie. Assessing creativity. In *Proceedings of the AISB01 Symposium on Artificial Intelligence and Creativity in Arts and Science*, pages 3–11, 2001.

[18] G. Ritchie, R. Manurung, H. Pain, A. Waller, R. Black, and D. O'Mara. A practical application of computational humour. In *Cardoso, A. & Wiggins, G. (Ed.). Proceedings of the 4th. International Joint Workshop on ComputationalCreativity, London, UK*, 2007.

[19] M. T. Wong and A. H. W. Chun. Automatic haiku generation using vsm. In *Proceeding of 7th WSEAS Int. Conf. on Applied Computer & Applied Computational Science (ACACOS '08), Hangzhou, China*, 2008.