A Survey on Intelligent Poetry Generation: Languages, Features, Techniques, Reutilisation and Evaluation

Hugo Gonçalo Oliveira
CISUC, Department of Informatics Engineering
University of Coimbra, Portugal
hroliv@dei.uc.pt

Abstract

Poetry generation is becoming popular among researchers of Natural Language Generation, Computational Creativity and, broadly, Artificial Intelligence. To produce text that may be regarded as poetry, computational systems are typically knowledge-intensive and deal with several levels of language. Interest on the topic resulted in the development of several poetry generators described in the literature, with different features covered or handled differently, by a broad range of alternative approaches, as well as different perspectives on evaluation, another challenging aspect due the underlying subjectivity. This paper surveys intelligent poetry generators around a set of relevant axis – target language, form and content features, applied techniques, reutilisation of material, and evaluation – and aims to organise work developed on this topic so far.

1 Introduction

Interest in the development of automatic methods for generating poetry dates back to the 1960s, even before computers were accessible to everyone. A commonly cited example is the creation of a large number of new poems by interchanging the lines in a set of poems, always respecting the relative position of each line within a stanza (Queneau, 1961). Early approaches to (so-called) experimental poetry applied a set of combinatory processes to existing poems, in order to generate new ones. This challenge was embraced by different groups, such as the Portuguese movement of Experimental Poetry (PO.EX, see e.g. the collection by Torres and Baldwin (2014)), or the French Atelier of Literature Assisted by Maths and Computers (ALAMO, see Oulipo (1981)). A notable work of the latter includes the *rimbaudelaires*, where the structure of a poem by Rimbaud is filled with vocabulary from Baudelaire's poems. At the time, interested people were mostly poets or researchers from the domain of humanities.

It was not until the turn of the millennium that this area started to get attention from the Computer Science community, mainly researchers in the domain of Artificial Intelligence (AI) and, specifically, Computational Creativity (Colton and Wiggins, 2012). Since the works of Gervás (2000) and Manurung (2003), the development of "intelligent" poetry generation systems has seen a significant increase. These systems are not limited to rewriting text in a poetic form. They are often knowledge-intensive natural language generation systems that deal with several levels of language (e.g. phonetics, lexical choice, syntax and semantics) to produce aesthetically-pleasing text with a creative value.

What makes this task more interesting is that some levels of language do not have to be strictly addressed. Writing poetic text does not have to be an extremely precise task (Gervás, 2000), as several rules, typically present in the production of natural language, need to be broken (Manurung, 2003). On the other hand, poetry involves a high occurrence of interdependent linguistic phenomena where features such as metre, rhyme, and the presence of figurative language play an important role. For instance, it might be ok to transmit a less clear message, in a trade-off for a pleasant sound given by a highly regular metre. In fact, the latter is easier to achieve by a

Proceedings of The 10th International Natural Language Generation conference, pages 11–20,
Santiago de Compostela, Spain, September 4-7 2017.

computer, while transmitting a clear message can be highly challenging, especially when constrained by the previous features.

Given the challenge involved and the range of possibilities, it is not surprising to see many poetry generators developed as serious research efforts where distinct approaches to tackle this common goal are explored, reported in scientific papers, with enough detail, and even compared to prior work. Besides the involved challenge, poetry generators can be useful for applications in different areas, such as education or electronic entertainment.

While not exactly an introduction to the topic, this paper surveys poetry generation systems according to relevant axis where different trends have emerged for the goal of generating poetry automatically – though not necessarily enough to cover every single detail in this topic. Instead of a full description of specific systems, it is organised around the identified axis and focuses on distinct ways they were handled by different systems. For those interested on the topic, this can be seen as a quicker reference for selecting suitable approaches, possibly tackling their limitations, or opt instead for alternative approaches, novel for poetry generation.

This paper is structured as follows: Section 2 enumerates the languages for which intelligent poetry generators have been developed. Section 3 overviews the most common poetic features considered by these systems. Section 4 focuses on the different approaches and resources that poetry generators resort to for selecting their content. Section 5 describes some of the AI techniques applied by these systems towards their goal. Section 6 is about the different degrees to which human-produced text is exploited or reused by different systems, for inspiration. Section 7 overviews distinct evaluation approaches applied to poetry generators. Given the subjectivity involved in poetry generation, not all surveyed systems have been evaluated, and most have only to a certain extent. Section 8 concludes this paper with some final remarks.

2 Languages

Poetry is an artistic expression of language. Humans have produced poetry in many languages and, due to their specificities, different languages happen to follow different poetic traditions, often focused on different forms. While the majority of poetry generation systems targets English and produces text in this language, there are systems of this kind in other languages, enumerated in this section.

Well-known early attempts to poetry generation included French (Queneau, 1961; Oulipo, 1981), but Spanish was one of the first languages where this topic was explored in the context of AI, and related issues were discussed (Gervás, 2000; Gervás, 2001). For Portuguese, another romance language, song lyrics have been automatically generated for a given melody (Gonçalo Oliveira et al., 2007), and poetry has been produced according to user-given structures that would set the number of lines, stanzas, syllables per stanza, or the rhyme pattern (Gonçalo Oliveira, 2012). In an effort to use the same architecture for generating poetry in different languages, the previous system was extended to cover also Spanish and English (Gonçalo Oliveira et al., 2017). Another poetry generator originally developed for English was also adapted to produce poetry in Spanish (Ghazvininejad et al., 2016).

Traditional eight-line Basque poems, aiming to be sung, have also been produced automatically (Agirrezabal et al., 2013). Although, as Portuguese and Spanish, Basque is spoken in the Iberian Peninsula, it has different origins and is significantly different from romance languages. Toivanen et al. (2012)'s system produced poetry in Finnish, another European language.

Asian languages have also been targeted, some of which with specific tonal and rhythm requirements in poetry generation. This includes the generation of song lyrics in Tamil (Ramakrishnan A et al., 2009; Ramakrishnan A and Devi, 2010), a phonetic language; ancient Chinese classic poetry (Yan et al., 2013; Zhang and Lapata, 2014; Yan, 2016), with strict tonal and rhythm requirements; follow-up lines in Bengali (Das and Gambäck, 2014), matching the rhythm of a user-given line; and poetry inspired by news articles, in Indonesian (Rashel and Manurung, 2014).

3 Form features

Despite all the levels of language involved in poetry, form is a key feature for, at the first glance, recognis-

ing the resulting text as poetic. Most common form-related features are, without a doubt, a regular metre and rhymes. When alone, both of them are quite straightforward to handle by computer programs, especially when compared with content features.

Metre is generally modelled with the number of syllables each line has, sometimes also considering the stress patterns (e.g. Manurung (2003), Gervás (2001), Tobing and Manurung (2015)), which indicate the position of the stressed syllables. Rhyme results from the repetition of certain sounds (e.g. in *great* and *mate*). End-rhymes, the most typical, occur when two lines end in the same sound. But some systems consider other kinds of rhyme, such as assonance or alliteration, which respectively involve the repetition of the same vowel or of a consonant sound throughout the poem.

For less phonetic languages, such as Portuguese (Gonçalo Oliveira et al., 2007) or Spanish (Gervás, 2001), it is often enough to design a set of orthography-based rules to handle metre and rhyme. For English, poetry generators (e.g. Manurung (2003), Colton et al. (2012), Tobing and Manurung (2015)) typically resort to a pronunciation dictionary for this purpose (e.g. CMU's[1]). Yet, automatic methods for the automatic scansion of poetry have also been developed (Agirrezabal et al., 2016).

Metre and rhymes are often organised according to a well-known poetry form and some systems are designed to produce only poems of specific forms. *Haikus* traditionally have 3 lines, respectively with 5, 7 and 5 syllables (Manurung, 2003; Netzer et al., 2009), but there are modern haikus with a different number (Wong and Chun, 2008). *Limericks* have five lines, with lines 1, 2 and 5 generally longer, and rhyme of the kind AABBA (Levy, 2001; Manurung, 2003). The *sonnet* is a classic form of poem with 14 lines, typically with 10-syllables each. Depending on the tradition, it might have different groupings, stress patterns and rhyming schemes, such as ABAB CDCD EFEF GG (Ghazvininejad et al., 2016). Spanish traditional forms (Gervás, 2000; Gervás, 2001) include the *romance*, lines of 8 syllables, where all even-numbered rhyme together; the *cuarteto*, a stanza with four 11-syllable lines, where the two outer lines rhyme together; and *tercetos encadenados*, stanzas of three 11-syllable lines with the pattern ABA BCB CDC... *Bertsolaritza* is a Basque traditional verse with metre and rhyme constraints, typically sung (Agirrezabal et al., 2013). The generation of classic Chinese poetry has focused mostly on *quartrains*, four lines of 5 or 7 characters with a rigid tonal pattern where two kinds of tones are interleaved, and a rhyme scheme where the majority of the lines in the same poem end with the same vowel, but not the same character (Yan et al., 2013; Zhang and Lapata, 2014; Yan, 2016).

The poetry form can be decided from the initial data (Gervás, 2000), while other systems generate poetry in more or less any form, depending on a user-provided template, which might be strictly structural (Gonçalo Oliveira, 2012) or a poem, possibly with some words stripped (Toivanen et al., 2014). There are also systems focused on generating song lyrics, which have less traditional forms, but where metre is key for matching the rhythm, while other features should still be present. These include melodies where stressed and weak beats are identified (Gonçalo Oliveira et al., 2007; Ramakrishnan A et al., 2009; Gonçalo Oliveira, 2015), pop songs (Barbieri et al., 2012), or rap (Malmi et al., 2016; Potash et al., 2015) where, besides rhyme, assonance is modelled as the repetition of vowel phonemes (e.g. in *raps* and *tax*).

4 Content features

Even though form ends up shaping content, it is not enough for a poem to simply follow a recognisable form of poetry. According to Manurung (2003), besides *poeticness*, poetic text should hold two other fundamental properties: it must obey linguistic conventions, prescribed by a given grammar and lexicon (*grammaticality*); and it must convey a conceptual message, meaningful under some interpretation (*meaningfulness*). To some extent, following syntactic rules is often a consequence of most of the surveyed approaches, as they rely on text fragments, lexical-syntactic patterns or language models acquired from human-produced text (see following sections). On the other hand, meaningfulness is less trivial to handle automatically. Given the involved challenge, it is not always explicitly consid-

[1] http://svn.code.sf.net/p/cmusphinx/code/trunk/cmudict/

ered and is often only softly satisfied, for instance, by using words that belong to the same semantic domain. This section describes how different poetry generators select their content in order to transmit a meaningful message or, at least, to be, as much as possible, semantically coherent.

Intelligent poetry generation systems often exploit a model of semantics, either a semantic knowledge base, or a statistical model of distributional semantics. The former is usually a more theoretical view on linguistic knowledge, where words are connected according to labelled relations, with different meanings. Poetry generators have used knowledge bases with verbs and their restrictions and ontological categories (Ramakrishnan A and Devi, 2010); semantic networks extracted from dictionaries, that go beyond synonymy and hypernymy, and cover other relations such as causation, property and others (Gonçalo Oliveira, 2012); WordNet, a lexical knowledge base (Colton et al., 2012; Agirrezabal et al., 2013; Tobing and Manurung, 2015); and ConceptNet, a common sense knowledge base (Das and Gambäck, 2014). Those have been used not only to restrict the generated words to a common semantic domain, but also for increasing the paraphrasing power, towards higher variation and better covering of different metres.

Distributional models of semantics target how language is actually used, in a collection of documents, and consider that words that occur in similar contexts have similar meanings. These include vector space models, either based on words (Wong and Chun, 2008; McGregor et al., 2016), also including word embeddings learned from collections of poems (Yan, 2016) or from Wikipedia (Ghazvininejad et al., 2016), or based on sentences (Malmi et al., 2016), both used to compute the semantic relatedness with the cosine similarity; or word associations (Netzer et al., 2009; Toivanen et al., 2012) which, according to some authors, capture relations in poetic text better than WordNet-like lexical knowledge bases.

In some systems, text is generated according to a grammar for handling syntax, possibly also considering semantic features (Manurung, 2003). In Gonçalo Oliveira (2012)'s system, the grammar is tightly related to the semantics, as each rule transmits a known semantic relation and can be instantiated with any pair of words sharing relations of that kind (e.g. *vehicle-car* or *fruit-mango*, for hypernymy).

Yet, in order to enable some kind of interpretation, the poem must actually be about something or, at least, be different for different stimuli, reflected in its content. Stimuli can be given in different forms, with different degrees of precision, namely: a list of semantic predicates (e.g. *love(John, Mary)*) (Manurung, 2003); one (Netzer et al., 2009; Toivanen et al., 2013; Ghazvininejad et al., 2016) or more (Wong and Chun, 2008; Gonçalo Oliveira, 2012; Zhang and Lapata, 2014; Yan, 2016) keywords that will, somehow, set a semantic domain and constraint the generation space; a line of text (Das and Gambäck, 2014) or a sequence of lines (Malmi et al., 2016) to be followed; a textual document, which can either be a single sentence with a message (Gervás, 2001), or a longer text from a blog (Misztal and Indurkhya, 2014) or newspaper (Díaz-Agudo et al., 2002; Colton et al., 2012; Rashel and Manurung, 2014; Toivanen et al., 2014; Tobing and Manurung, 2015; Gonçalo Oliveira and Alves, 2016).

In order to extract meaningful information to be used in the poem, different systems process the input document differently. For instance, Toivanen et al. (2014) acquire novel associations from the document (e.g. *bieber* and *alcohol*, in opposition to *pop* and *star*), identified by contrast with well-known associations. Tobing and Manurung (2015) extract dependency relations from the document and use them to constrain the generated poem. They argue that, though not a genuine semantic representation, dependency relations are a useful abstraction of the text and end up conveying its semantics. In fact, some dependency relations include semantic relations (e.g. *agent-of*, *subject-of*, *object-of*). A final example (Gonçalo Oliveira and Alves, 2016) extracts concept maps from the input document, and uses them as a semantic network.

Towards an improved interpretation, Colton et al. (2012)'s system produces natural language commentaries for each generated poem, providing some generation context. A similar feature is presented by Gonçalo Oliveira and Alves (2016) or Gonçalo Oliveira et al. (2017). In this case, semantic relation instances explaining the connection between the input keywords and the words used can be provided

either in raw format or, if a grammar exists for this purpose, in natural language.

Additional semantic features captured by poetry generators include sentiment (Gervás, 2000; Colton et al., 2012; Gonçalo Oliveira et al., 2017), which typically involves exploiting a polarity lexicon; or emotion (Misztal and Indurkhya, 2014), in this case achieved with the help of WordNet Affect.

Figurative language is often implicitly present as a consequence of reusing material from human-produced poetry, but its presence can also be explicitly handled, for instance, by exploiting similes mined from Google n-grams (Colton et al., 2012). Veale (2013) points out the importance of content-features and presents a system more relaxed on form but heavily influenced by figurative language. More precisely, similes (e.g. *politicians are crooks*) are exploited for generating metaphors (e.g. *he is a crook*) and conceptual blends (e.g. *sweet silence*).

Poetry generation systems handle a broad range of features both at the formal and at the content level. Dealing with so many constraints may actually turn out to be computationally impractical (see e.g. Tobing and Manurung (2015)). Yet, this also depends on the techniques adopted for handling all the constrains, surveyed in the following section.

5 Artificial Intelligence Techniques

Early poetry generators (e.g. Queneau (1961) or Oulipo (1981)) relied heavily on combinatory processes applied to a set of human-created poems. On the other hand, intelligent poetry generation systems consider semantics when selecting content and take advantage of computational techniques that add value to the generation process, with a more efficient exploration of the space of possible generations, also enabling to handle a larger number of features, often towards a predefined intention, and sometimes resulting in poems with higher novelty. This section enumerates some of those techniques, borrowed from the domain of AI.

The technique of Case-Based Reasoning exploits past solutions for solving new similar problems, in a four-step approach (*retrieve*, *reuse*, *revise*, *retain*). In the scope of poetry generation (Gervás, 2001; Díaz-Agudo et al., 2002), it has been instantiated as follows: *retrieve* vocabulary and line examples that suit fragments of a poem draft; *reuse* the part-of-speech (POS) structure of the example lines for producing new lines and combine them with the words in the vocabulary; present the resulting draft to the user, for *revision*; perform a linguistic analysis of the revised poems and *retain* it for further generations.

Chart Parsing is a known technique that employs dynamic programming for parsing text according to a context-free grammar. Chart Generation, used by some poetry generation systems (Manurung, 1999; Manurung, 2003; Tobing and Manurung, 2015; Gonçalo Oliveira, 2012), is the inverse of chart parsing. Given a grammar, a lexicon, and a meaning (e.g. as a set of predicates), chart generation produces all syntactically well-formed texts that convey the meaning. Charts store complete generated constituents (inactive edges) as well as incomplete (active edges), with dotted rules marking constituent portions yet to be generated.

Poetry composition can be seen as an incremental task, where initial drafts go through several iterations, each ideally better than the previous, until the final poem. The application of evolutionary algorithms to this task (Levy, 2001; Manurung, 2003) is thus natural. The basic idea is to generate an initial population of poems by a simple method, and then evolve it through several generations, towards more suitable poems, assessed by a fitness function that considers a set of relevant features for poetry. Changes in the population are obtained by the application of crossover and mutation operators. Crossover creates new poems from two other poems in the population. This can be achieved by adopting the syntax of the former but the words or the rhyme of the latter (Levy, 2001), or by swapping parts of the former with parts of the latter (Manurung, 2003). Mutation may involve the replacement of some words in all the poem, only in a certain line, or changing the rhyme (Levy, 2001). It may also consist of adding, deleting or changing contents of the poem, possibly considering the target semantics (Manurung, 2003).

Given the number of constraints involved in poetry generation, it is also natural to have this problem formulated as a Constraint Satisfaction approach (Toivanen et al., 2013; Rashel and Manurung, 2014). For this purpose, a constraint satisfaction solver explores the search space and produces

solutions that match the input properties (how poems can be like), represented as predicates that indicate the poem structure and the vocabulary words to be used. Different constraints and their types can be set for different generations. Hard constraints are mandatory (e.g. number of lines, syllables per line), while soft constraints are optional (e.g. rhymes).

Language models have been used to generate poetic text, constrained by both a target style and a predefined form. These include Markov models (Barbieri et al., 2012) and models based on Deep Neural Networks (DNNs), including Recurrent Neural Networks (RNNs). Given a sequence of words, a RNN was used to predict the next word in rap lyrics (Potash et al., 2015). Or given the line history, RNNs can be used for generating new lines incrementally, considering their respective phonetics, structure and semantics (Zhang and Lapata, 2014; Yan, 2016). There may be one neural network (NN) for selecting the structure of lines and another for guiding the generation of single words within a line. Towards better poeticness, Yan (2016) goes further and adds poem refinement iterations to the previous process. The RNN language model may also be guided by a Finite-State Acceptor that controls rhyme and metre (Ghazvininejad et al., 2016). Malmi et al. (2016) use a DNN and the RankSVM algorithm to predict the next full line, from a knowledge base of human-produced lyrics, considering rhyme, structure and semantic similarity.

Support Vector Machines (SVMs), trained in a poetry corpus, were also used to predict follow-up lines with certain syllabic and rhyming properties (Das and Gambäck, 2014). And classic NNs were also used to measure the fitness of poems generated by an evolutionary approach (Levy, 2001). In the latter case, the NN was trained on human judgments of creativity in a selection of *limericks*, half by humans and another half randomly generated.

Misztal and Indurkhya (2014) adopted a Multi-Agent approach where a set of artificial experts, focused on a particular aspect of poetry generation, interact by sharing results on a blackboard. Experts can contribute with words matching a given topic or emotion (*word-generating*), arrange words in the common pool into phrases (*poem-making*), or select the best solutions according to given constraints and heuristics (*selection experts*), among others.

Poetry generation has also been tackled as a Generative Summarization framework that incorporates poetic features as constraints to be optimised (Yan et al., 2013). Candidate poems are retrieved for a set of keywords, they are segmented into constituent terms and clustered given their semantics. Lines that conform the structural constraints, each using terms from the same cluster and with some correlation, are then selected. Suitable term replacements are finally made iteratively, in order to improve structure, rhyme, tonality and semantic coherence.

6 Reutilisation of Materials

To avoid the generation of poems completely from scratch, most poetry generators take shortcuts and rely on human-created text, usually poems, for inspiration. Different systems exploit the inspiration set differently, generally for the acquisition of useful knowledge or guidelines that will simplify how certain features (e.g. form, syntax or even figurative language) are handled, and also to help modelling the produced poems towards recognisable poetry. This is also reflected on how the inspiration contents are reused in the produced poems.

Some systems acquire full lines or fragments from human-created poems (Queneau, 1961), rap lyrics (Malmi et al., 2016), blog posts (Wong and Chun, 2008), or tweets (Charnley et al., 2014), and recombine them in new poems, considering features such as metre, rhymes, presence of certain words or semantic similarity. On the one hand, these solutions minimize the issues of dealing with syntax and do not require an underlying generation grammar or template. On the other hand, from the point of view of novelty, they are poor, as lines from known texts can be spotted in the middle of the produced poems. If precautions are not taken, this can even lead to licensing issues.

Other systems operate on templates to be filled with new words. Templates can be handcrafted and cover variations of similes and key phrases from newspapers (Colton et al., 2012), or they can be extracted automatically from text. The latter kind may be based on full poems or on single lines, where some words are replaced by others with similar grammatical features, possibly further constrained on metre, rhyme, POS or semantics. For instance,

full poems can have content words stripped (Toivanen et al., 2012; Toivanen et al., 2013; Toivanen et al., 2014; Rashel and Manurung, 2014). Systems that reuse full fragments may also include an additional step where certain words are replaced, in order to better satisfy the target features (Yan et al., 2013).

Line templates can be extracted fragments where a semantic relation must be held between stripped words (e.g. *dark <x> on a dangerous <y>*, where *partOf(x, y)*) (Gonçalo Oliveira, 2012; Gonçalo Oliveira and Alves, 2016; Gonçalo Oliveira et al., 2017), or sequences of POS-tags (e.g. *NN-NN-JJ-VB*) (Gervás, 2000; Gonçalo Oliveira et al., 2007; Agirrezabal et al., 2013). To some extent, POS-tag templates can be seen as flat grammars.

A different way of exploiting the inspiration set involves acquiring a lexicon with the most common words used in this set, and using it for guiding the generation process (Wong and Chun, 2008).

7 Evaluation

Poetry generation is becoming a mature research field, which is confirmed by several works that go beyond the production and exhibition of a few interesting poems that, to some extent, match the target goals. Despite the subjective aspect that makes poem evaluation far from trivial, it is more and more common to explore different ways for assessing both the obtained results and the generation process.

Claiming that the intended audience of poetry consists of people, the evaluation of computer generated poetry has often resorted to human judges, who assess produced poems according to a set of predefined dimensions. For instance, although, to some extent, the properties of *poeticness*, *grammaticality* and *meaningfulness* can be validated by the methods applied (Manurung, 2003; Misztal and Indurkhya, 2014), they can also be assessed by the observation of the obtained results. Having this in mind, some researchers (Yan et al., 2013; Das and Gambäck, 2014; Zhang and Lapata, 2014; Yan, 2016) evaluated the output of their system based on the opinion of human judges on a set of produced poems, who answered questionnaires designed to capture the aforementioned properties. Still relying on human opinions, other authors took conclusions on the quality of their results with questions that rated slightly different dimensions, though with some overlap. Those include the typicality as a poem, understandability, quality of language, mental images, emotions, and liking (Toivanen et al., 2012); or structure, diction, grammar, unity, message and expressiveness (Rashel and Manurung, 2014).

Some of the previous systems ended up conducting a Turing test-like evaluation, where the scores of the systems produced by their poems were compared to those for human-created poems (Netzer et al., 2009; Toivanen et al., 2012; Agirrezabal et al., 2013; Rashel and Manurung, 2014). Despite also relying on human evaluation, other researchers compared poems produced only by their systems but using different parameters or strategies (Gervás, 2000; Gonçalo Oliveira et al., 2007; Barbieri et al., 2012; Yan et al., 2013; McGregor et al., 2016); or poems produced by other systems with a very similar purpose (Zhang and Lapata, 2014; Yan, 2016).

Established methods to evaluate human creativity, from the psychology domain, have also been proposed to assess computational creativity, including automatically generated poetry. van der Velde et al. (2015) present a map of words related to creativity (e.g. unconventional, spontaneitiy, imagination, planning, craftmanship, art), obtained from an association study. These words were clustered and may be used to define relevant dimensions for evaluating creativity, for instance, in a poem, and in its creation process. Another study employing methods from psychology (Lamb et al., 2016) resorted to human experts for rating the creativity of poems, some written by humans and others generated automatically, by different creative systems. Judges were not informed of this and, despite some consensus on the best and worst poems, they disagreed on the remaining, which made the authors unsure on the suitability of their approach for computer-generated poetry.

There has been a huge discussion on the suitability of the Turing test for evaluating computational creativity approaches (Pease and Colton, 2011). The main criticism is that it is focused on the resulting products and not on the involved creative process, which encourages the application of simpler processes, some of which might be merely concerned with tricking the human judge into thinking their outputs were produced by a human.

The FACE descriptive model (Colton et al., 2011)

has been proposed to evaluate the creative process and was used in the evaluation of poetry generation systems (Colton et al., 2012; Misztal and Indurkhya, 2014). To be assessed positively by this model, a creative system must create a concept (C), with several examples (E), include an aesthetic measure (A) for evaluating the concept and its examples, and provide framing information (F) that will explain the context or motivation of the outputs. Yet, other experiments (McGregor et al., 2016) suggest that framing, which can be provided as a natural language commentary, does not make a big difference in human assessment of the creativity, meaningfulness or general quality of computer-generated poems.

In many systems, the evaluation of certain features is part of the process, as it happens for Misztal and Indurkhya (2014)'s automatic experts, or with the large number of systems that assess the metre, rhymes and other properties of produced texts during generation time. Few approaches have tried to evaluate poetry generation systems or their results automatically, and these often tackled less subjective dimensions of poems. Those include the application of metrics typically used in the scope of automatic summarization and machine translation, such as ROUGE, to access the performance of a poetry generator based on Generative Summarization (Yan et al., 2013); or BLEU, to assess the ability to generate valid sequences of lines (Zhang and Lapata, 2014; Yan, 2016). ROUGE was also used to assess variation in poems generated by the same system with the same parameters (Gonçalo Oliveira et al., 2017). Moreover, the perplexity of the learned language model has been compared to human-produced poetry with a similar style (Zhang and Lapata, 2014; Yan, 2016); the average cosine similarity of the lines in automatically-created *haikus* has been compared to the same value for awarded *haikus* by humans, to conclude that semantic coherence is similar (Wong and Chun, 2008); and Pointwise Mutual Information, computed on Wikipedia, has been used to measure the association between seeds and words effectively used (Gonçalo Oliveira, 2015; Gonçalo Oliveira et al., 2017).

Other systems focused on measuring the novelty of their results, especially those that reuse more material or try to model an artist's style. The main goal is to generate poems that share some similarities with the inspiration set, but are different from any existing poem. Potash et al. (2015) compute the cosine similarity between the automatically generated lines and the original lines by the target artist (the lower cosine, the higher the novelty). To compute similarity, the number of rhyming syllables is divided by the total number of syllables (rhyme density) and the result is compared to the same number for the lyrics of the target artist. Rhyme density is also computed by Malmi et al. (2016) and, to some extent, by Gonçalo Oliveira et al. (2017). In the latter case, it is given by the number of lines with an end-rhyme divided by the total number of lines.

8 Final discussion

Many computational systems that tackle the complexity of poetry generation from an AI perspective have been developed and thoroughly described in scientific literature. We can thus say that this topic, with interest for the communities of Computational Creativity and Natural Language Generation, is today an established research field.

Intelligent poetry generators were surveyed in this paper, around a set of relevant axis where alternative approaches have been explored. Poetry has been automatically generated in different languages and forms, considering different sets of features, and through significantly different approaches. Poetry generators have been developed with different goals and intents, each with their stronger and weaker points (though this is out of the scope of the current paper), which adds to the subjectivity involved in the evaluation of poetry, even for humans. This explains the broad range of approaches explored for poetry generation, though not one can be said to be better than the others. Nevertheless, many researchers are making progress towards the evaluation of the generation process and its impact on the obtained results.

We do hope that this paper can be used as a future reference for people working or considering to work on poetry generation. Many of the approaches described have room for further exploration, not to mention that several alternative features, constraints or AI techniques are still left to be explored in this scope. Some of which might result in surprisingly interesting results, with potential applications in education or entertainment.

References

Manex Agirrezabal, Bertol Arrieta, Aitzol Astigarraga, and Mans Hulden. 2013. POS-tag based poetry generation with wordnet. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 162–166, Sofia, Bulgaria, August. ACL Press.

Manex Agirrezabal, Iñaki Alegria, and Mans Hulden. 2016. Machine learning for metrical analysis of english poetry. In *Proceedings of 26th International Conference on Computational Linguistics: Technical Papers*, pages 772–781, Osaka, Japan, December. COLING 2016 Organizing Committee.

Gabriele Barbieri, François Pachet, Pierre Roy, and Mirko Degli Esposti. 2012. Markov constraints for generating lyrics with style. In *Proceedings of 20th European Conference on Artificial Intelligence (ECAI)*, volume 242 of *Frontiers in Artificial Intelligence and Applications*, pages 115–120. IOS Press.

John Charnley, Simon Colton, and Maria Teresa Llano. 2014. The FloWr framework: Automated flowchart construction, optimisation and alteration for creative systems. In *Proceedings of 5th International Conference on Computational Creativity, ICCC 2014*, Ljubljana, Slovenia, June.

Simon Colton and Geraint A. Wiggins. 2012. Computational creativity: The final frontier? In *Proceedings of 20th European Conference on Artificial Intelligence (ECAI 2012)*, volume 242 of *Frontiers in Artificial Intelligence and Applications*, pages 21–26, Montpellier, France. IOS Press.

Simon Colton, John Charnley, and Alison Pease. 2011. Computational creativity theory: The face and idea descriptive models. In *Proceedings of the 2nd International Conference on Computational Creativity*, page 90–95, México City, México, April.

Simon Colton, Jacob Goodwin, and Tony Veale. 2012. Full FACE poetry generation. In *Proceedings of 3rd International Conference on Computational Creativity*, ICCC 2012, pages 95–102, Dublin, Ireland.

Amitava Das and Björn Gambäck. 2014. Poetic machine: Computational creativity for automatic poetry generation in Bengali. In *Proceedings of 5th International Conference on Computational Creativity*, ICCC 2014, Ljubljana, Slovenia, June.

Belén Díaz-Agudo, Pablo Gervás, and Pedro A. González-Calero. 2002. Poetry generation in colibri. In *Proceedings of 6th European Conference on Advances in Case-Based Reasoning (ECCBR 2002)*, pages 73–102, London, UK. Springer.

Pablo Gervás. 2000. WASP: Evaluation of different strategies for the automatic generation of spanish verse. In *Proceedings of AISB'00 Symposium on Creative & Cultural Aspects and Applications of AI & Cognitive Science*, pages 93–100, Birmingham, UK.

Pablo Gervás. 2001. An expert system for the composition of formal Spanish poetry. *Journal of Knowledge-Based Systems*, 14(3–4):181–188.

Marjan Ghazvininejad, Xing Shi, Yejin Choi, and Kevin Knight. 2016. Generating topical poetry. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1183–1191, Austin, Texas, November. ACL Press.

Hugo Gonçalo Oliveira, F. Amílcar Cardoso, and Francisco Câmara Pereira. 2007. Tra-la-Lyrics: an approach to generate text based on rhythm. In *Procs. of 4th International Joint Workshop on Computational Creativity*, pages 47–55, London, UK. IJWCC 2007.

Hugo Gonçalo Oliveira. 2012. PoeTryMe: a versatile platform for poetry generation. In *Proceedings of the ECAI 2012 Workshop on Computational Creativity, Concept Invention, and General Intelligence*, C3GI 2012, Montpellier, France, August.

Hugo Gonçalo Oliveira and Ana Oliveira Alves. 2016. Poetry from concept maps – yet another adaptation of PoeTryMe's flexible architecture. In *Proceedings of 7th International Conference on Computational Creativity*, ICCC 2016, Paris, France.

Hugo Gonçalo Oliveira, Raquel Hervás, Alberto Díaz, and Pablo Gervás. 2017. Multilanguage extension and evaluation of a poetry generator. *Natural Language Engineering*, page in press. (online since June 2017).

Hugo Gonçalo Oliveira. 2015. Tra-la-lyrics 2.0: Automatic generation of song lyrics on a semantic domain. *Journal of Artificial General Intelligence*, 6(1):87–110, December. Special Issue: Computational Creativity, Concept Invention, and General Intelligence.

Carolyn Lamb, Daniel Brown, and Charles Clarke. 2016. How digital poetry experts evaluate digital poetry. In *Proceedings of 7th International Conference on Computational Creativity*, ICCC 2016, Paris, France.

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

Eric Malmi, Pyry Takala, Hannu Toivonen, Tapani Raiko, and Aristides Gionis. 2016. Dopelearning: A computational approach to rap lyrics generation. In *Proceedings of 22nd SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 195–204, San Francisco, CA, USA, August. ACM Press.

Hisar Manurung. 1999. A chart generator for rhythm patterned text. In *Proceedings of 1st International Workshop on Literature in Cognition and Computer*.

H. M. Manurung. 2003. *An evolutionary algorithm approach to poetry generation*. Ph.D. thesis, University of Edimburgh, Edimburgh, UK.

Stephen McGregor, Matthew Purver, and Geraint Wiggins. 2016. Process based evaluation of computer generated poetry. In *Proceedings of the INLG 2016 Workshop on Computational Creativity in Natural Language Generation*, pages 51–60, Edinburgh, UK, September. ACL Press.

Joanna Misztal and Bipin Indurkhya. 2014. Poetry generation system with an emotional personality. In *5th International Conference on Computational Creativity, ICCC 2014*, Ljubljana, Slovenia, 06/2014.

Yael Netzer, David Gabay, Yoav Goldberg, and Michael Elhadad. 2009. Gaiku: generating haiku with word associations norms. In *Proceedings of the Workshop on Computational Approaches to Linguistic Creativity*, CALC'09, pages 32–39. ACL Press.

(Association) Oulipo. 1981. *Atlas de littérature potentielle*. Number vol. 1 in Collection Idées. Gallimard.

Alison Pease and Simon Colton. 2011. On impact and evaluation in computational creativity: A discussion of the turing test and an alternative proposal. In *Proceedings of the 3rd AISB symposium on AI and Philosophy, University of York, York, UK*.

Peter Potash, Alexey Romanov, and Anna Rumshisky. 2015. GhostWriter: Using an LSTM for automatic rap lyric generation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2015, pages 1919–1924, Lisbon, Portugal. ACL Press.

R. Queneau. 1961. *100.000.000.000.000 de poèmes*. Gallimard Series. Schoenhof's Foreign Books, Incorporated.

Ananth Ramakrishnan A and Sobha Lalitha Devi. 2010. An alternate approach towards meaningful lyric generation in tamil. In *Proceedings of the NAACL HLT 2010 2nd Workshop on Computational Approaches to Linguistic Creativity*, CALC'10, pages 31–39. ACL Press.

Ananth Ramakrishnan A, Sankar Kuppan, and Sobha Lalitha Devi. 2009. Automatic generation of Tamil lyrics for melodies. In *Proceedings of Workshop on Computational Approaches to Linguistic Creativity*, CALC'09, pages 40–46. ACL Press.

Fam Rashel and Ruli Manurung. 2014. Pemuisi: A constraint satisfaction-based generator of topical Indonesian poetry. In *Proceedings of 5th International Conference on Computational Creativity*, ICCC 2014, Ljubljana, Slovenia, June.

Berty Chrismartin Lumban Tobing and Ruli Manurung. 2015. A chart generation system for topical metrical poetry. In *Proceedings of the 6th International Conference on Computational Creativity, Park City, Utah, USA*, ICCC 2015, Park City, Utah, USA, Jun.

Jukka M. Toivanen, Hannu Toivonen, Alessandro Valitutti, and Oskar Gross. 2012. Corpus-based generation of content and form in poetry. In *Proceedings of the 3rd International Conference on Computational Creativity*, ICCC 2012, pages 211–215, Dublin, Ireland, May.

Jukka M. Toivanen, Matti Järvisalo, and Hannu Toivonen. 2013. Harnessing constraint programming for poetry composition. In *Proceedings of the 4th International Conference on Computational Creativity*, ICCC 2013, pages 160–167. The University of Sydney.

Jukka M. Toivanen, Oskar Gross, and Hannu Toivonen. 2014. The officer is taller than you, who race yourself! using document specific word associations in poetry generation. In *5th International Conference on Computational Creativity, ICCC 2014*, Ljubljana, Slovenia, 06/2014.

Rui Torres and Sandy Baldwin, editors. 2014. *PO.EX: Essays from Portugal on Cyberliterature and Intermedia by Pedro Barbosa, Ana Hatherly, and E.M. de Melo e Castro*. West Virginia University Press / Center for Literary Computing.

Frank van der Velde, Roger A Wolf, Martin Schmettow, and Deniece S Nazareth. 2015. A semantic map for evaluating creativity. In *Proceedings of the 6th International Conference on Computational Creativity June*, page 94.

Tony Veale. 2013. Less rhyme, more reason: Knowledge-based poetry generation with feeling, insight and wit. In *Proceedings of the 4th International Conference on Computational Creativity*, pages 152–159, Sydney, Australia.

Martin Tsan Wong and Andy Hon Wai Chun. 2008. Automatic haiku generation using VSM. In *Proceeding of 7th WSEAS International Conference on Applied Computer & Applied Computational Science*, ACACOS '08, Hangzhou, China.

Rui Yan, Han Jiang, Mirella Lapata, Shou-De Lin, Xueqiang Lv, and Xiaoming Li. 2013. I, Poet: Automatic Chinese poetry composition through a generative summarization framework under constrained optimization. In *Proceedings of 23rd International Joint Conference on Artificial Intelligence*, IJCAI'13, pages 2197–2203. AAAI Press.

Rui Yan. 2016. i, Poet: Automatic poetry composition through recurrent neural networks with iterative polishing schema. In *Proceedings of 25th International Joint Conference on Artificial Intelligence*, IJCAI 2016, pages 2238–2244, New York, NY, USA, July. IJCAI/AAAI Press.

Xingxing Zhang and Mirella Lapata. 2014. Chinese poetry generation with recurrent neural networks. In *Proceedings of 2014 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2014, pages 670–680, Doha, Qatar, October. ACL Press.