Towards finer-grained interaction with a Poetry Generator

Hugo Gonçalo Oliveira
hroliv@dei.uc.pt

Tiago Mendes
tjmendes@student.dei.uc.pt

Ana Boavida
aboavida@dei.uc.pt

CISUC, Department of Informatics Engineering
University of Coimbra, Portugal

Abstract

PoeTryMe is a poetry generation system that produces poems autonomously, from a set of initial parameters. After using a simplified version of this system, creative writers and other interested people identified some issues and expressed their wish to make changes in the resulting poems or to interact with the system and take part in the creative process. This paper illustrates some of the issues on generated poems and reports on a recent effort towards providing alternative ways of using PoeTryMe and to meet the previous suggestions. Some functionalities were made available via a web API, which can now be exploited by other systems. Those include the generation of single lines, or the retrieval of related words, where additional constrains can be made on the number of syllables, sentiment or rhyme. On the top of this API, a co-creative interface has been developed. It enables users to start from scratch, from an existing poem, or to generate a new draft.

1 Introduction

PoeTryMe [6] is a platform for automated poetry generation. So far, it has been adapted to produce poetry in different forms, languages and from different stimuli. PoeTryMe generates a poem from a set of initial parameters, such as the poetry form, the language, a set of seeds or a surprise factor. An implemented generation strategy is then followed to select the most suitable lines to fill the form, generated with the help of a semantic network and a generation grammar. The result is a sequence of semantically-coherent lines, using the seeds or words related to them, grouped according to the given poetry form, with the corresponding number of syllables, and often with rhymes. This is done in an autonomous fashion, with no additional user interaction nor input besides the initial parameters.

Yet, after using a simplified version of PoeTryMe, several users, including musicians and poets, expressed their interest to be more involved in PoeTryMe's creation process. Some even confessed to having generated several poems, keeping only some of the best lines, and created a new improved poem from this manual procedure. Although the obtained results are generally ok, especially if generation goes through more iterations in the search for the most suitable lines, there are always aspects that may be improved, not to mention that assessing the quality of poetry is a subjective task and may diverge from user to user.

In order to meet the users feedback, we decided to provide direct access to some of the main functionalities required for poetry generation, through a web API. This was possible due to the modular architecture of PoeTryMe, which already decouples the process of poetry generation in several tasks, performed independently.

In: A.M. Olteteanu, Z. Falomir (eds.): Proceedings of ProSocrates 2017: Symposium on Problem-solving, Creativity and Spatial Reasoning in Cognitive Systems, at Delmenhorst, Germany, July 2017, published at http://ceur-ws.org

Functionalities now available through the API include the generation of single lines from a seed, with a target number of syllables, or the retrieval of words related to another, where additional constraints can be made on the number of syllables, sentiment or rhyme. To some extent, this is in line with the vision of a Creative Web, where creative tools are deployed as web services [20], and enables third-party applications to interact with PoeTryMe, not only for the generation of a single poem, but also to customise when each of the available functionalities is performed, possibly combining them in alternative ways.

One of such applications is Co-PoeTryMe, a web-based tool that enables to use PoeTryMe co-creatively. Users may start from scratch, with their own words and lines, they can import an existing poem, or they can generate a newl draft poem, which should help them to overcome the 'blank page syndrome'[1] [14]. In any case, lines and words may then be switched or replaced with new ones, following certain user-defined constraints, hopefully towards a better poem, more aligned with their intentions.

The remaining of this paper starts with a brief introduction to related topics, including poetry generation, co-creativity, poetry and co-creativity, and finally creativity-support interfaces. After this, PoeTryMe is overviewed, followed by an enumeration of some identified limitations, with illustrative examples. Together with the user feedback, the previous limitations motivate most functionalities that were made available, described next. Before concluding, Co-PoeTryMe is revealed and its main features are briefly described.

2 Related Work

Poetry generation is a popular task in the scope of Computational Creativity [3]. During the last 15 years, several systems for this purpose were developed, using a wide range of artificial intelligence techniques (e.g. [4, 17, 2, 19, 6, 18, 16], see [10] for a short survey). The majority of those systems generates poetry autonomously, from initial stimuli, either provided by the user or triggered by some event, and outputs the results without any human intervention during the generative process.

A different kind of systems enable the collaboration of humans and computational agents in the production of creative artefacts. This can be done through alternating co-creativity, when the computer performs exactly the same tasks as the human, though in different turns; or through task-divided co-creativity, when the tasks performed by the computer are different than those by the human [15]. For instance, in the poetry domain, some tasks should be easier for a computer program – e.g., finding words that rhyme or with a certain number of syllables, which, to some extent, can be made with the help of a set of rules and / or a pronunciation lexicon – , while other tasks would suit the human better – e.g., evaluating the aesthetics, which can be very subjective; or conveying a non-trivial meaning, which might require access to much knowledge organised for this purpose.

We identified existing computational systems that enable the collaboration between man and machine in the production of poems [14, 16, 21], though with different interfaces and available functionalities. The Poetry Machine [14] is presented as an interactive tool developed on the top of a poetry generation system [19]. From a set of user-given keywords, the system generates initial draft lines, which may be further changed by the user or by the system, in equal turns. During the creation process, the user may ask for additional lines or words, while the new fragments produced by the system will automatically adapt to the user's modifications. jGnoetry is a web application[2] that enables the generation of poems from a collection of texts provided by the user, who may also select the poetry form from a list of predefined forms or set its own through sequences of syllables organised in lines. The resulting poem is presented in such a way that the user may select words to keep in further iterations and generate new text for the unselected words. Deep Beat [16] is a rap lyrics generation system with a web interface[3] where the users can write some lines, provide a set of words to appear, and ask the system for the suggestion of new lines, possibly rhyming. Each suggested line appears after a picture of its original author, because they were collected from human-created rap lyrics. Another lyrics-writing supporting system is LyriSys [21], where the user sets a musical structure, chooses a topic for each block or writes part of the lyrics, and the system generates lyrics for the remaining blocks, following the given structure and within the selected topic. After this, users can still revise the lines they are not happy with, possibly replacing them with alternative candidates.

From a similar perspective, Misztal and Indurkhya's [18] poetry generation system could also be seen as co-creative, but the collaboration does not involve humans, only computational agents with different expertises.

[1]The 'blank page syndrome' refers to when a writer opens a blank page and cannot or takes too long to start writing because there are no words on the page

[2]`http://www.eddeaddad.net/jgnoetry/`

[3]`http://deepbeat.org/`

More precisely, agents with different responsibilities, such as dealing with emotion, word generation, poetic aspects, or selecting the best solutions, interact in a common workspace (blackboard), towards the production of a poem.

A different kind of creativity platform is not co-creative nor creative on its own, but allows users to combine different more or less creative services through a web interface, in the development of novel creative workflows, which can be tested right away. Such platforms include ConCreTeFlows [22] and FloWr [1] and have been used for poetry generation, among other tasks.

In order to be integrated in such platforms, creative systems must somehow be in line with the vision of a Creative Web, where creative tools are deployed as web services [20] that may be used by third-party applications. Concerning poetry generation, the previous vision is further discussed by Gervás [5], who argues that abstractions of the various functionalities involved in a poetry generation system should be available as services that may be later invoked by other systems. This enables the development of different systems that would, nevertheless, share some modules, thus requiring less effort to develop. As for the co-creative systems, they must communicate with their user interfaces and allow to run different steps involved in poetry generation at will, instead of always going through the full process. So, even if these steps are not always available as services, modularity should be present and, in most cases, each module could potentially become available as an independent service.

3 PoeTryMe and its Modular Architecture

PoeTryMe [6] is a computational platform, originally designed to test different settings in the process of poetry generation, with a focus on the exploited knowledge resources, which could be indirectly assessed this way. For this reason, PoeTryMe has a modular architecture that enables not only the independent development of each module, but also to test different settings of input parameters and to study their impact in the resulting poems, with reduced effort. PoeTryMe's architecture has two core modules: the Line Generator produces semantically-coherent lines with the help of a semantic network and a generation grammar, with textual patterns for rendering semantic relations, given their type; and the Generation Strategy retrieves lines on a semantic domain, from the Line Generator, and selects which will be used in the poem. Additional modules can be seen as complementary. A more extensive description of this architecture is found elsewhere [11].

Among other parameters, users may define the rules of a generation grammar, the underlying semantic network, the poetry form, the set of seed words, the polarity lexicon and the transmitted sentiment. Developers may additionally reimplement some of the modules and reuse the others. Each different setting consists of a new instantiation of PoeTryMe. So far, PoeTryMe has been instantiated to produce poetry in different forms, including song lyrics [7], and in different languages, originally Portuguese [6], and later also Spanish and English [11], given different stimuli, such as Twitter trends [9] or concept maps extracted from text [13].

A limited version of PoeTryMe can currently be used through the TryMe web interface[4] [8], which generates poems given a language, one of the predefined forms, a list of seeds and a surprise factor. Poems are generated following a generate-and-test strategy, also used in most instantiations of PoeTryMe: lines are generated, one after another, until the metre and rhyme constraints are met or a certain number of generations is reached. The generation strategy cannot be changed through TryMe, nor can additional parameters, such as the semantic network, the generation grammar, turning the seed expansion on, or the desired polarity. Yet, this interface enabled the identification of most limitations discussed in the following paragraphs, together with the *Poeta Artificial*[5] Twitter bot [9], which continuously generates Portuguese poems inspired by current trends. We should nevertheless note that some limitations can be minimised if PoeTryMe is used with its full capabilities. For instance, increasing the number of generations, not possible through the interface, increases the chance of rhymes in a trade-off of longer generation time.

4 Identified Limitations of PoeTryMe

The main limitation of PoeTryMe is that the meaning and intention of the poems is not clearly-defined. Given seed words constrain the semantic network, so that each line uses them or semantically-related words. Each relation can be rendered as text following a set of patterns in the generation grammar, which apply for every relation of the same kind. Poems will thus be richer for larger semantic networks, not only in terms of words and relation instances, but with a varied range of relation types. The generation grammar also plays an important

[4]TryMe section in `http://poetryme.dei.uc.pt`
[5]Check `https://twitter.com/poetartificial`

role, because it should contain several different renderings for each relation type, and for lines with different number of syllables.

Although the aforementioned approach enables the generation of semantically-coherent lines, in the strategies implemented so far, they are generated independently. Therefore, although the connection to the seeds provides some consistence and the poem is indeed on a certain topic (check a previous evaluation [11]), the sequence of ideas is not always the best. Also, the same word might have different meanings, which, in some cases, might be interesting, but may also shift the poem semantics. One the other hand, although this feature is not available with the TryMe interface, the system can explain the semantic connection between the selected words and their connection to the seeds, which might be helpful to understand the underlying meaning. This limitation can be further minimised if the semantic network is replaced by one specifically-tailored for the target domain, as in previous work [13], where it was extracted from a given Wikipedia article, but the generation grammar would also have to be tuned, regarding the extracted relations and, especially, the user intention.

The following poem, generated with the seeds *poetry* and *interaction*, illustrates the possible results of PoeTryMe.

and interaction aggressions
no more poetry expressions
tag that poetry too loud
action says it, stage says it

Each line has seven syllables and two of them rhyme, but their sequence is not the most logical. Semantic consistence is aided by the relations rendered in each line, which involve the seeds or related words, namely: coHyponymOf(*interaction*, *aggression*), coHyponymOf(*expression*, *poetry*), domainOf(*tag*, *poetry*), hyponymOf(*action*, *stage*). The last instance is indirectly connected to *interaction*, through hypernymOf(*interaction*, *action*), which is possible because the poem was generated with a surprise factor greater than 0.

This leads us to another limitation of PoeTryMe: it might be tricky to select the surprise factor. This parameter sets the probability of selecting words that, in the semantic network, are more than one level further to the seeds. More precisely, a high surprise factor increases variation in vocabulary by considering more words that are not directly connected with the seeds. On the one hand, a low surprise might result in a poem where every line uses one seed, thus not so interesting. See, for instance, the following example, generated with the seeds *problem* and *solving*, with surprise set to 0.

fall like a dead riddle problem
we got solving and locations
solving write determinations
with problem and pitfall on?

On the other hand, a higher surprise might result in a poem where the connection to the domain is harder to find, such as the following, generated with the seeds *spatial* and *cognition* and a surprise of 0.1 – meaning that every relation instance connected to a word that is connected to a seed has a 10% probability of being used by the Line Generator.

if you locate what you place
the mind doubters of her embrace
ears were nasty, heads were hot
chair, moon, and places forgot

Despite an indirect connection with the seed words, none of them ends up being used. As a probability, the impact of this factor is highly influenced by the number of relation instances involving the seeds and the words directly related to them – e.g. 10% of all the words related to *person* or *place* can be a much higher number than 10% of all the words related to *solving* or *computational*, because the former are nodes with a higher degree in the network. In fact, when the number of relations involving each seed is not balanced, the resulting poem might use only words related to the seed that is involved in more relation instances. This happens, for instance, in the following poem, generated with *computational* and *creativity*, where every line uses the latter or related words (namely, coHyponymOf(*hands*, *creativity*)) and the former is just forgotten:

design leads to creativity
creativity plane aptitudes
while the stormy hands do right
creativity spin, flight

Besides making the poems potentially more interesting, this issue can be minimised if the initial seeds are automatically expanded with a small set of relevant words, obtained with the PageRank algorithm (as in previous work [7, 11]). Yet, again, this feature is not available in the TryMe interface.

A final limitation is related to the grammar renderings. One assumption behind PoeTryMe is that all relation instances of the same type can be expressed by the same textual pattern, changing only the involved words. For instance, "*the <x> of the <y>*" should hold for every pair of words, such that $partOf(x, y)$. However, in order to avoid time-consuming work for handcrafting this kind of grammars, they ended up being automatically extracted from human-created poems and song lyrics, which enabled the creation of larger and richer grammars, but also harder to control. Therefore, poems occasionally present grammatical inconsistencies, odd syntactic constructions, or words from a different semantic domain, that are a fixed part of the pattern. See, for instance, the following poem, generated with the seeds *human*, *creative*, and *cognition*:

a imaginative creative sea
a insertion where contents are free
license plate with mankind human on it
filled with joy... human becomes mitt

In the first two lines, the determiner *a* should instead be *an*, because it precedes words starting with a vowel. Furthermore, words such as *sea* or *license plate* are respectively part of the pattern used and, although might accidentally result in something interesting, do not have a strong connection with the seeds. Another issue is the odd construction *mankind human.*

5 Extending PoeTryMe's API

Since TryMe became available, it has been tested by different people and, together with the outputs of the *Poeta Artificial* Twitter bot [9], the previously discussed limitations were either identified or confirmed. In this process, some users pointed out items that would improve the system, often expressing their will to change the resulting poem. For instance, in some cases: they preferred a different line order; some lines did not end in rhyme, or did not match the target metre exactly; punctuation was missing or there was a grammatical inconsistence; the syntax or the semantics was bizarre; they were looking for lines with a deeper meaning; they just wanted to change a word or a line for no specific reason. Some users even confessed to have generated several poems, possibly with different parameters, in order to obtain more suitable lines or words, which they would then manually organise as a new poem. This was our main motivation for providing an alternative way of using PoeTryMe.

Before this work, PoeTryMe's API already had a REST API with a single endpoint:

- `http://poetryme.dei.uc.pt:8080/PoetrymeWeb/rest/poetry`

Used by TryMe and by ConCreTeFlows [22], this endpoint enabled the generation of a full poem from a small set of parameters, namely:

- Language: `lang=[en|pt|es]`
- Form: `form=[id of the form]` (either from a pre-defined list, also including song lyrics; or a string $n \times m$, where n is the number of lines and m is the number of syllables)
- Seeds: `words=[comma-separated list of words]`
- Surprise: `surp=[0-1]`

For instance, the following URL returns a sonnet in Portuguese, using the seeds *criatividade* and *computador*, with a surprise of 0.01:

```
http://poetryme.dei.uc.pt:8080/PoetrymeWeb/rest/poetry?lang=pt&form=sonnet&seeds=computador+
criatividade&surp=0.01
```

Towards finer-grained interaction with PoeTryMe, the API was extended with additional endpoints for performing lower-level functionalities, namely:

- Generation of a single line, given a list of seeds and a surprise factor, selected from the best *n* lines generated, according to a target number of syllables:

  `http://poetryme.dei.uc.pt:8080/PoetrymeWeb/rest/poetry/line`

  - Language: `lang=[en|pt|es]`
  - Seeds: `seeds=[comma-separated list of words]`
  - Surprise: `surp=[0-1]`
  - Target number of syllables: `nsyl=[1-n]`
  - Maximum generations: `bestof=[1-n]`

- Retrieve a set of words related to a target word. Relation can be semantic, same number of syllables, same rhyme, or a combination of the previous. Resulting words might be further constrained with a target polarity:

  `http://poetryme.dei.uc.pt:8080/PoetrymeWeb/rest/poetry/words`

  - Language: `lang=[en|pt|es]`
  - Word: `word=[word]`
  - Relation: `rel=[id for relation type]` (supported types are: `synon` for synonymy, `anton` for antonymy, `hyper` for hypernym, `hypon` for hyponymy, `cohyp` for co-hyponymy, `other` for any other type, or `any` for any type)
  - Target rhyme: `rhyme=[word with the target rhyme]`
  - Target number of syllables: `syl=[word with the target number of syllables]`
  - Polarity: `pol=[-1,0,1]`

- Score a piece of text for a target poetry form. This will use the same metric as PoeTryMe, which means that poems that perfectly match the metre will be score with 0, with (negative) bonus for each pair of rhyming lines, and a (positive) penalty for each syllable out-of-metre:

  `http://poetryme.dei.uc.pt:8080/PoetrymeWeb/rest/poetry/score`

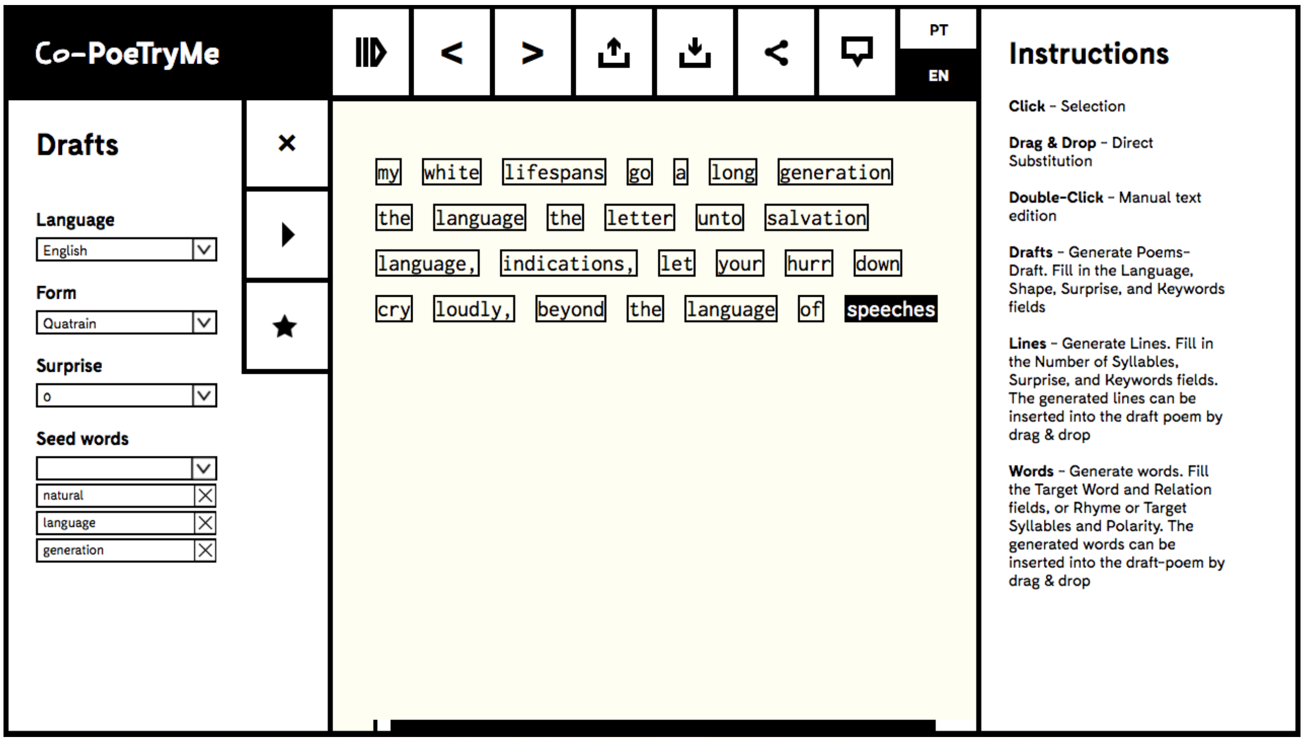
  - Language: `lang=[en|pt|es]`
  - Form: `form=[id of the form]` (same possibilities as for the poem generation URL)
  - Text: `text=[full text]` (lines split with a \n)

The new functionalities can be exploited by different systems, which may use them for different approaches to poetry generation or alternative purposes. In the next section, a use case of this API is presented: a co-creative web interface for PoeTryMe that uses these functionalities on demand.

6 Co-PoeTryMe: a Co-Creative Interface for PoeTryMe

Co-PoeTryMe [12] is a web-based tool, mostly developed in JavaScript, that enables the human user to interact with PoeTryMe, hopefully towards better poems, or more in line with their intentions. Interaction starts with a draft poem, which may be fully generated by the system, written by the user from scratch, or imported from an existing text file. After this, the user may manually edit parts of the poem, select words or full lines.

Co-PoeTryMe has several visual modules, but they are not all visible at the same time. Figure 1 depicts the modules for poem Edition (centre), Draft generation (left) and Instructions (right) in the current prototype. Once the draft has content, the Edition module becomes visible. The user can double-click to edit the full poem, a line, or a word manually, and can also drag-and-drop words or lines to switch them. The Drafts module has buttons for enabling the generation of a new draft that will replace the current one (play-like button), and to score the current draft based on the metrics and the presence of rhyme, according to the selected poetry form (star-like button). The top module, always visible, has utility buttons for selecting the application language (English or Portuguese), showing or hiding tooltips, importing or exporting a draft, sharing in social networks, undo, redo, as well as a tool for visualising the changes made from the initial draft to its current state.



Language: English
Form: Quatrain
Surprise: 0
Seed words: natural, language, generation

my white lifespans go a long generation
the language the letter unto salvation
language, indications, let your hurr down
cry loudly, beyond the language of speeches

Instructions

Click - Selection

Drag & Drop - Direct Substitution

Double-Click - Manual text edition

Drafts - Generate Poems-Draft. Fill in the Language, Shape, Surprise, and Keywords fields

Lines - Generate Lines. Fill in the Number of Syllables, Surprise, and Keywords fields. The generated lines can be inserted into the draft poem by drag & drop

Words - Generate words. Fill the Target Word and Relation fields, or Rhyme or Target Syllables and Polarity. The generated words can be inserted into the draft-poem by drag & drop

Figure 1: Co-PoeTryMe's prototype: poem edition and word generation.

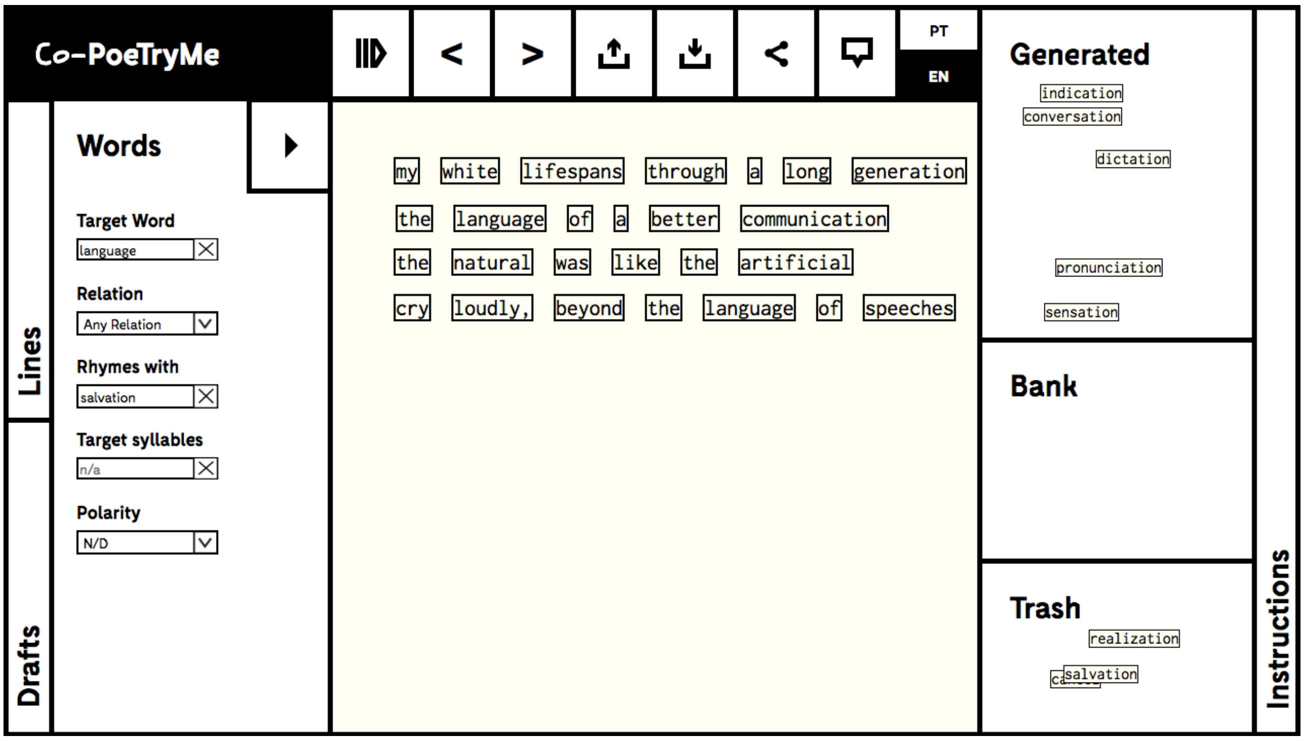
When a word is clicked, word-related tools are shown instead of the Drafts and Instructions modules, as depicted in figure 2. On the left, the Words module enables the retrieval of words with certain features, such as a semantic relation, rhyme or the same number of syllables as a selected word, possibly with a given polarity. Retrieved words appear in a cloud, located in the Generated Words module, on the right-hand side. They can be drag-and-dropped to be added to the poem, they can replace words in the poem, or they can be moved to the Word Bank module, for future utilisation. Replaced words are moved to the Trash module. At each word generation action, triggered by the play-like button, the words in the Generated module are lost, while words in the Bank and in the Trash remain available for future inclusion in the poem. In figure 2, words related to *language* and rhyming with *salvation* were retrieved.

When a complete line is clicked, the Lines module is shown instead of the Words or the Drafts module. The Lines module enables the generation of new lines and, similarly to the words, it also enables a Generated, a Bank and a Trash module, specific for lines, on the right-hand side. Figure 3 shows the line-related modules while, in the center, the changes made by the user in the original draft are illustrated. Currently, the represented changes cover words and lines switched and words added, removed or replaced. Symbols inspired by document revision tasks are applied, and the text produced by the human has a distinct typography than text produced automatically. Besides a static visual representation of the changes, changes can be animated from the beginning to the current state of the draft, through a play-like button that appears in the bottom of the Edition module. The static representation may also be exported as an image and drafts may be directly shared in social networks.

7 Conclusion and Further Work

This paper described how the suggestions of several users of PoeTryMe, a poetry generation system, were followed to enable a finer-grained and more user-driven interaction with this system, while also minimising some of its identified limitations. The modular architecture of PoeTryMe eased the extension of its public API with additional endpoints for performing sub-tasks of poetry generation, such as the generation of single lines or the retrieval of related words. This enables the exploitation of specific modules of PoeTryMe by external systems such as Co-PoeTryMe, a co-creative tool that enables the user to collaborate with PoeTryMe in the composition of new poems.

In the future, we are planning to make more functionalities available through the API, such as the contextualization of lines according to the underlying semantic network. We are still working on the development of



Target Word: language

Relation: Any Relation

Rhymes with: salvation

Target syllables: n/a

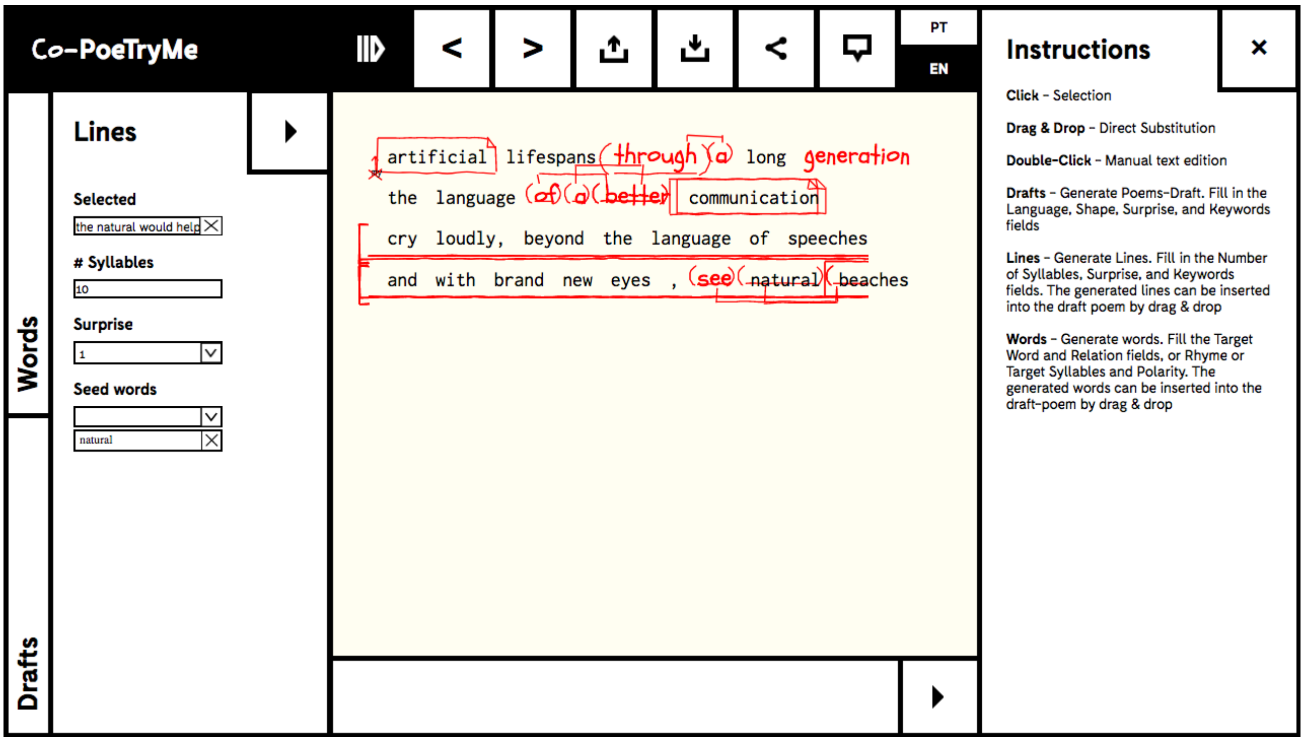
Polarity: N/D

my white lifespans through a long generation
the language of a better communication
the natural was like the artificial
cry loudly, beyond the language of speeches

Generated: indication, conversation, dictation, pronunciation, sensation

Trash: realization, salvation

Figure 2: Co-PoeTryMe's prototype: poem edition and word generation.



Lines

Selected: the natural would help

# Syllables: 10

Surprise: 1

Seed words: natural

artificial lifespans ~~through~~ ~~a~~ long generation
the language ~~of~~ ~~a~~ ~~better~~ communication
cry loudly, beyond the language of speeches
and with brand new eyes , ~~see~~ ~~natural~~ beaches

Instructions

Click - Selection

Drag & Drop - Direct Substitution

Double-Click - Manual text edition

Drafts - Generate Poems-Draft. Fill in the Language, Shape, Surprise, and Keywords fields

Lines - Generate Lines. Fill in the Number of Syllables, Surprise, and Keywords fields. The generated lines can be inserted into the draft poem by drag & drop

Words - Generate words. Fill the Target Word and Relation fields, or Rhyme or Target Syllables and Polarity. The generated words can be inserted into the draft-poem by drag & drop

Figure 3: Co-PoeTryMe's prototype: visualising changes.

Co-PoeTryMe, especially on the usability and design aspects. Usability tests are currently being conducted and will hopefully provide us useful feedback on aspects to improve and directions to take. New functionalities may also be added in a near future (e.g., score the metre of poems, according to a given form). Co-PoeTryMe is accessible online, from a link in PoeTryMe's website, at: `http://poetryme.dei.uc.pt/`.

References

[1] John Charnley, Simon Colton, Maria Teresa Llano, and Joseph Corneli. The FloWr online plat-form: Automated programming and computational creativity as a service. In *Proceedings of 7th International Conference on Computational Creativity (ICCC 2016)*, Paris, France, 2016. Sony CSL.

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

[3] Simon Colton and Geraint A. Wiggins. 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, 2012. IOS Press.

[4] Pablo Gervás. 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, 2000.

[5] Pablo Gervás. Deconstructing computer poets: Making selected processes available as services. *Computational Intelligence*, 33(1):3–31, 2017.

[6] Hugo Gonçalo Oliveira. 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 2012.

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

[8] Hugo Gonçalo Oliveira. Poe, now you can TryMe: Interacting with a poetry generation system. In *Proceedings of the Demonstration Session of PROPOR'2016, 12th International Conference on Computational Processing of the Portuguese Language*, 2016.

[9] Hugo Gonçalo Oliveira. O Poeta Artificial 2.0: Increasing meaningfulness in a poetry generation twitter bot. In *Proceedings of the 2nd Workshop on Computational Creativity in Natural Language Generation*, CC-NLG, page (in press), Santiago de Compostela, Spain, 2017. ACL Press.

[10] Hugo Gonçalo Oliveira. A survey on intelligent poetry generation: Languages, features, techniques, reutilisation and evaluation. In *Proceedings of the 10th International Conference on Natural Language Generation*, INLG 2017, page (in press), Santiago de Compostela, Spain, 2017. ACL Press.

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

[12] Hugo Gonçalo Oliveira, Tiago Mendes, and Ana Boavida. Co-PoeTryMe: a co-creative interface for the composition of poetry. In *Proceedings of the 10th International Conference on Natural Language Generation (Demo Session)*, INLG 2017, page (in press), Santiago de Compostela, Spain, 2017. ACL Press.

[13] Hugo Gonçalo Oliveira and Ana Oliveira Alves. 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, 2016.

[14] Anna Kantosalo, Jukka M. Toivanen, Ping Xiao, and Hannu Toivonen. From isolation to involvement: Adapting machine creativity software to support human-computer co-creation. In *Proceedings of 5th International Conference on Computational Creativity, Ljubljana, Slovenia, June 10-13, 2014*, 2014.

[15] Anna Kantosalo and Hannu Toivonen. Modes for creative human-computer collaboration: Alternating and task-divided co-creativity. In *Proceedings of 7th International Conference on Computational Creativity (ICCC 2016)*, Paris, France, 2016. Sony CSL.

[16] Eric Malmi, Pyry Takala, Hannu Toivonen, Tapani Raiko, and Aristides Gionis. Dopelearning: A computational approach to rap lyrics generation. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 195–204, 2016.

[17] H. M. Manurung. *An evolutionary algorithm approach to poetry generation.* PhD thesis, University of Edimburgh, Edimburgh, UK, 2003.

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

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

[20] Tony Veale. Creativity as a web service: A vision of human and computer creativity in the web era. In *Creativity and (Early) Cognitive Development: A Perspective from Artificial Creativity, Developmental AI, and Robotics, Papers from the 2013 AAAI Spring Symposium, Palo Alto, California, USA, March 25-27, 2013*, volume SS-13-02. AAAI Press, 2013.

[21] Kento Watanabe, Yuichiroh Matsubayashi, Kentaro Inui, Tomoyasu Nakano, Satoru Fukayama, and Masataka Goto. LyriSys: An interactive support system for writing lyrics based on topic transition. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces*, IUI '17, pages 559–563, New York, NY, USA, 2017. ACM.

[22] Martin Żnidaršič, Amílcar Cardoso, Pablo Gervás, Pedro Martins, Raquel Hervás, Ana Oliveira Alves, Hugo Oliveira, Ping Xiao, Simo Linkola, Hannu Toivonen, Janez Kranjc, and Nada Lavrac. Computational creativity infrastructure for online software composition: A conceptual blending use case. In *Proceedings of 7th International Conference on Computational Creativity (ICCC 2016)*, Paris, France, 2016. Sony CSL.