A Preliminary Study for Literary Rhyme Generation based on Neuronal Representation, Semantics and Shallow Parsing

Luis-Gil Moreno-Jiménez[1], **Juan-Manuel Torres-Moreno**[1], **Roseli S. Wedemann**[2]

[1]Laboratoire Informatique d'Avignon – Avignon Université, Avignon, France

[2]Inst. de Matemática e Estatística – Univ. do Estado do Rio de Janeiro, RJ Brazil

luis-gil.moreno-jimenez@univ-avignon.fr,

juan-manuel.torres-moreno@univ-avignon.fr, roseli@ime.uerj.br

Abstract. *In recent years, researchers in the area of Computational Creativity have studied the human creative process proposing different approaches to reproduce it with a formal procedure. In this paper, we introduce a model for the generation of literary rhymes in Spanish, combining structures of language and neural network models The results obtained with a manual evaluation of the texts generated by our algorithm are encouraging.*

1. Introduction

For many years, research in Artificial Intelligence (AI) has directed efforts towards automating processes to perform specific academic, industrial or economic tasks for society. However, the investigation and development of procedures for the automation of human artistic and creative processes has not had as much attention due to the complexities involved in these activities. Procedures developed for these purposes involve mathematical-computational methods designed to process and learn from a large quantity of digital data, so as to detect patterns in order to simulate the creative process (CP), as explained by Boden in [Boden 2004].

In this paper, we introduce a model for the generation of rhymes with literary components. Our proposal is based on findings detailed in [Moreno-Jiménez et al. 2020a], where Automatic Text Generation (ATG) techniques are combined with neural network (NN) based models, such as the *Word2vec* algorithm [Mikolov et al. 2013b], for the generation of literary texts. In Section 2, we present some of the literature regarding literary text generation, focusing on methods related to this paper. In Section 3, we explain the RIMAX model used to generate the rhyming words. In Section 4, we describe the corpora used for the learning phase of our models. Then, in Section 5, we explain the methodology implemented for the generation of rhymes. We show some experiments and examples in Section 6, as well as the results of evaluations conducted by humans. Finally, we present conclusions and propose possible future works in Section 7.

2. Related Work

There has been much interest and work in the area of ATG with different and interesting goals, regarding the different types of texts. Many of the proposed algorithms are based on neural networks. In [Kiddon et al. 2016], the authors generate coherent text using a recurrent neural network (RNN) and a *neural checklist model*. Their RNN predicts the best context from a list of keywords. Another RNN approach is proposed in [Clark et al. 2018] for generating narrative text, such as fiction or news stories. Entities mentioned in the text are represented by vectors, which are updated as the text generation proceeds as they represent different contexts and guide the RNN in determining the vocabulary to be retrieved in order to generate a narrative.

We note that there are other ATG techniques, such as *text realization* that creates text in a human language, *e.g.* English or French, from a syntactic representation [Molins and Lapalme 2015]. Oliveira has written a survey of work treating the automatic generation of poetry [Oliveira 2017], and presents his own method for generating poems based on the use of templates (*canned text*) in [Oliveira and Cardoso 2015]. Another work based on canned text is presented in [Agirrezabal et al. 2013], which generates strophes of verses for Basque poetry. In [Zhang and Lapata 2014], a RNN was proposed for the generation of Chinese poetry based on learning of known text structure.

3. Semantic Rhyme

RIMAX is the first automatic system for detecting semantic rhymes in Spanish [Urrea and Torres-Moreno 2019]. It contains the following ingredients: 1) a rhyming dictionary, 2) the set of definitions of those rhymes and 3) a strategy to measure semantic proximity. This procedure can be applied to different romance languages, although we have chosen the Spanish language spoken in Mexico, given the availability of some useful resources and tools, such as the Dictionary of Mexican Spanish (DEM)[1] and the Rhyming Dictionary [Medina Urrea 2018].

Rhyming dictionaries gather words according to rhyming patterns. *Consonant* rhymes share ending sequences of vocalic and consonant sounds and *assonant* rhymes share similar vowel sounds. These two classes are thus based on pronunciation features, not on writing patterns. Also, since consonance and assonance depend on the stressed syllable, words which end with a stressed syllable are grouped together, those whose stressed syllable is the next to last appear together, and so on[2]. In this paper we have used the nomenclature of the DEM to automatically generate a phonological transcription.

3.1. Rhyme Ranking by Definition Similarity

Online dictionaries offer useful and simple advantages such as ranking and ordering of results. From a language perspective, it is interesting that text mining techniques can be applied to accomplish this. In fact, text similarity measures can be used to determine how similar word definitions are, i. e. measuring definition similarity.

Let D be a dictionary containing the set of defined words w and the set of definitions d. Since a word may have several senses, let d_{ij} be the jth definition of word w_i in D. Similarly, let d'_{kl} be the lth definition of word w_k. Also, let $\vec{v_{ij}}$ and $\vec{v'_{kl}}$ be vectors where the frequencies of lemmatized or ultrastemmized [Torres-Moreno 2012] content words of definitions d_{ij} and d'_{kl} are stored. Then, the similarity between d_{ij} and d'_{kl} can then be measured using the well-known quantity called *cosine similarity measure*, s_c.

[1]*Diccionario del español de México*, `https://dem.colmex.mx/`.

[2]For example, the penultimate syllables of the following Spanish words are the stressed syllables: *angula, chula, mula, chamula*. So these words should appear together in a rhyming dictionary.

In order to find semantic rhymes, each member of the rhyming set of word x will be weighed according to how similar its definition is to that of x, using the similarity measurement s_c. Hence, given a query word, consonance and assonance lists are generated and ordered by the calculated similarity among definitions. The program RIMAX allows us to select some parameter values for experiments.

4. Corpus

The corpus **MegaLite-Es** was used to train our model. It consists of 5 075 literary documents (mainly books) in Spanish. This corpus can be useful for different NLP tasks. The documents of **MegaLite-Es** were obtained from different personal collections and, for copyrights reasons, the distribution of the original documents is not possible. Instead of this, in [Moreno-Jiménez and Torres-Moreno 2021] the authors propose some alternative resources.

4.1. Corpus Structure

The 5 075 documents in **MegaLite-Es** were written by 1 336 Spanish-speaking authors and official translations from languages other than Spanish. The documents represent different literary genres such as plays, poems, tales, essays, etc. We thus consider that this corpus is suitable for training *Word2vec* models.

The original documents, in heterogeneous formats[3] were processed to be converted into *utf8* encoded documents. A segmentation process divided the texts into sentences, corresponding to regular expressions, using a tool developed in PERL 5.0. Some undesirable data like: mutilated words, strange symbols and an unusual disposition of paragraphs could not be treated, although these situations are usual when dealing with this kind of corpora. Some characteristics of **MegaLite-Es** are detailed in Table 1.

Table 1. Characteristics of MegaLite-Es corpus (M = 10^6 and K = 10^3).

MegaLite-Es	Sentences	Tokens	Characters	Authors
Overall	15 M	212 M	1 265 M	1 328
Avg per document	3 K	41.8 K	250 K	–

MegaLite-Es has the advantage of being very extensive and suitable for automatic learning. It has, however, the disadvantage that many of its sentences consist of general language, without literary elements (stylized vocabulary or literary figures). However, these sentences often allow for fluent reading and provide the necessary links between the ideas expressed in the text, although they could imply some noisy results. As numbers identifying pages, chapters, sections or index could imply errors in the detection of sentences during segmentation, a manual process was performed to remove this undesirable data, although these errors may occur in a linguistic corpus with unstructured text.

5. Text Generation Model

In this section, we describe the model we have proposed for the generation of literary rhymes. The model consists of two steps described as follows.

[3]pdf, txt, html, doc, docx, odt, etc.

5.1. First Step: Canned Text Method

We implemented a *Canned Text* method, which has the advantage of being efficient for syntactic analysis in ATG tasks [van Deemter et al. 2005], to generate grammatical templates named Partially Empty Grammatical Structures (PGSs). Each PGS is composed of Part-of-Speech (POS) tags[4] and function words[5]. The POS tags are retrieved by Freeling [Padró and Stanilovsky 2012]. PGSs are created from a template set, called *TempSet*, that consists of sentences selected manually from the **MegaLite-Es** corpus, according to the following rules.

- Each sentence must express a concrete idea.
- Each sentence must have a length N, such that $5 \leq N \leq 10$.
- The sentence should contain at least three lexical words[6].

For generating rhymes, the process begins by selecting two original sentences $f1$ and $f2$ from *TempSet* with length N, $5 \leq N \leq 10$. Sentences $f1$ and $f2$ must satisfy two additional conditions: (*1*) both sentences must finish with a lexical word; (*2*) the lexical words finishing the sentences must have the same grammatical inflection. These sentences are analyzed with FreeLing to detect lexical words that are replaced by POS tags. We concentrate on lexical words because they provide the most meaningful information in a text [Bracewell et al. 2005]. Function words are retained in the sentence, as these are useful for maintaining the grammatical coherence and we therefore do not change them.

The idea is to generate artificial sentences from "human" sentences, respecting their grammatical structure and substituting only the lexical words by words with the same linguistic inflection but a different meaning. This technique of text generation is well-known as homo-syntax. In contrast to the paraphrase that keeps the same meaning between the original and the generated texts and changes the grammar, homo-syntax seeks to generate a new text with a different meaning than the original text, although with the same grammatical structure. In Fig. 1, we show an illustration of the proposed model. The filled boxes represent function words, whereas the empty boxes represent the lexical words that are replaced by POS tags. Once the pair of sentences has been transformed into a PGS, it will be further changed by the procedure of the second step.

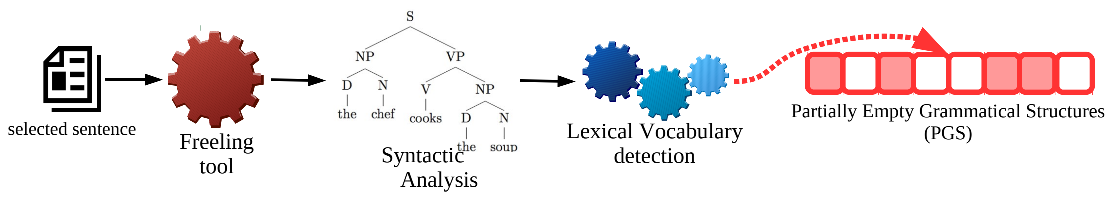

Figure 1. First step: Canned Text Method

5.2. Second step: Vocabulary selection

In this step, the POS tags in the PGSs are replaced by a vocabulary selected with the *Word2vec*[7] model. In a trained *Word2vec* model, each word, j, in the vocabulary is represented by a vector $\vec{L}_j$ with numerical valued elements. This allows for the implementation

[4]A POS tag indicates the part of speech grammatical category of a word.

[5]Prepositions, pronouns, auxiliary verbs, or conjunctions.

[6]Verbs, adjectives, nouns and adverbs

[7]*Word2vec* belongs to a group of ANN models, used to produce embeddings [Bengio et al. 2013].

of different mathematical procedures such as, for example, interpreting a semantic relation between a pair of words by calculating the cosine similarity between their vectors. These numerical vector representations are called *embeddings*.

The hyper-parameters configured for the *Word2vec* training were: *Iterations* = **10** (the number of training epochs over the **MegaLite-Es** corpus), *Minimum count* = **3** (the minimum number of times that a word must appear in the corpus to be included in the model's vocabulary), *Vector size* = **100** (the dimension of vectors, the *embeddings*) and *Window size* = **5** (the radius of adjacent words that will be related to the current word within a sentence, during the training phase of the model). We trained the model following the skip-gram procedure [Mikolov et al. 2013a]. Using the **MegaLite-Es** corpus for training, a trained model of 346 616 *Embeddings*[8] is obtained.

5.2.1. Word2vec Model

For the replacement, we used the analogical reasoning task introduced in [Mikolov et al. 2013a]. This reasoning consists of considering the relation between words, e. g. "France", "Paris", "Spain" and a missing word x. We suppose that "France", "Paris" and "Spain" are words that belong to the vocabulary of a corpus **CorpA** that was used to train *Word2vec*, and therefore, $\vec{Paris}, \vec{France}$, and $\vec{Spain}$ are the corresponding vectors associated to these words after training, respectively. The word x is then determined by finding a vector $\vec{x}$ associated to a word in **CorpA**, such that $\vec{x}$ is closest to $\vec{y} = \vec{Paris} - \vec{France} + \vec{Spain}$, according to the cosine similarity between $\vec{y}$ and $\vec{x}$ (see Eq. (2)). This specific example is considered to have been answered correctly, if $\vec{x}$ is the vector corresponding to "Madrid" in the vocabulary of **CorpA**. For the replacement of POS tags, we consider the three following words, their embedding vectors and Eq. (1),

Q: the context given by the user as a single query word,
O: the original word in $f1$ or $f2$ that is replaced by the POS tag,
A: the word adjacent to O on the left, in sentence $f1$ or $f2$, if it exists,

$$\vec{y} = \vec{A} - \vec{O} + \vec{Q}\,, \tag{1}$$

where $\vec{y}$ is the vector which we will use to choose the $M = 4\ 000$ closest embeddings. We rank the M embeddings in a list $\mathscr{L}$, by calculating the cosine similarity between the j^{th} embedding, $\vec{L_j}$, and $\vec{y}$,

$$\theta_j = \cos(\vec{L_j}, \vec{y}) = \frac{\vec{L_j} \cdot \vec{y}}{||\vec{L_j}|| \cdot ||\vec{y}||} \quad 1 \leq j \leq M. \tag{2}$$

$\mathscr{L}$ is ranked according to decreasing θ_j.

If we are replacing the first POS tag, then $A = None$, so we only compute $\vec{y} = \vec{O} + \vec{Q}$. For example, for the *Query* word *love* and the sentence: *I play the guitar*, we will replace the verb *play* and the noun *guitar*. Starting by the verb *play*, we compute $\vec{y} = \vec{play} + \vec{love}$ to get the ranked list $\mathscr{L}$. Some examples of returned embeddings are: *to like, to role, enchanting* and *abandon*. This list is then used in the analysis performed by the language model based on bigrams.

[8]Term used for the numerical representation of words for NLP, typically in the form of a real-valued vector that encodes the meaning of the word.

5.2.2. Language Model Analysis (Bigrams)

This step consists of calculating the conditional probability of a word, given a preceding word, that is

$$P(w_n|w_{n-1}) = \frac{P(w_n \wedge w_{n-1})}{P(w_{n-1})} \, . \tag{3}$$

Each bigram in the **MegaLite-Es** corpus has been detected and computed in this way. As a result, we have a new list of bigrams, LB. The bigrams are composed only by lexical and function words, ignoring punctuation, numbers and symbols. For each element in $\mathscr{L}$, we configure two bigrams, as b_1 and b_2, where:

- b_1 is the adjacent word to the left of O in $f1$ or $f2$, concatenated with the current analyzed word, L_j, and
- b_2 is the current analyzed word L_j concatenated with the adjacent word to the right of O, in $f1$ or $f2$.

We calculate the arithmetic mean, bm, of the frequencies of occurrence of b_1 and b_2 in LB. If O is the first (last) word in the sentence, we do not calculate any mean, and bm will be simply equal to the frequency of b_1 (b_2). The process is repeated for the M elements in $\mathscr{L}$. The values of bm are combined with the cosine similarities for each L_j, to re-rank $\mathscr{L}$ as

$$\theta_j = \frac{\theta_j + bm_j}{2} \, , \quad 1 \leq j \leq M \, . \tag{4}$$

Finally, we take the word at the top of the list as the chosen candidate to substitute O. The idea is to select the word that is semantically most similar to $\vec{y}$, based on the analysis accomplished with *Word2vec*, and maintain coherence of the text obtained with guidance of the language model. The process is repeated for each word in $f1$ and $f2$, except when we replace the last word in the second sentence, $w2_L$. To replace $w2_L$, we present the word that substituted $w1_L$ in $f1$ as input to RIMAX. RIMAX returns a ranked list LR with consonant and assonant rhymes related to $w1_L$. A score, LR_w is attributed to each word, w in LR, corresponding to a semantic similarity measure, which results from the semantic-phonetic analysis performed by a hybrid, automatic and manual process. The scores are normalized in the interval $[0-1]$. The scores in LR are combined with the scores in $\mathscr{L}$. For this, an average score is calculated for each pair of elements LR_w and L_w, where L_w corresponds to the element of $\mathscr{L}$ with the information referring to w.

$$\theta_w = \frac{\theta_w + LR_w}{2} \, , \qquad \forall w \in LR \, . \tag{5}$$

The words in LR that do not exist in $\mathscr{L}$ are also considered, and since LR_j is divided by 2, this strategy allows us to prioritize the elements contained in both lists. The new values in $\mathscr{L}$ are then processed with the language model as already described. Then we take the element in $\mathscr{L}$ in the first place, which contains the best semantic and coherent rhyme. Finally, a morphological analysis is performed once again with FreeLing, in order to convert the selected word into the correct inflection specified by the POS tag, if necessary. For that, we carry out conjugations and genre or number conversions.

The result is a newly generated pair of phrases that does not exist in the **MegaLite-Es** corpus, where $w2_L$ must rhyme with $w1_L$. The model is illustrated in Fig. 2, where the two PGSs (for $f1$ and $f2$) can be appreciated at the top of the illustration. Both structures are sending inputs to the *Word2vec* model, which receives Q, A and O in order to generate the list $\mathscr{L}$ with the vocabulary related to the inputs. It can be observed that the RIMAX module outputs its result to $f2$. RIMAX receives $w1_L$ and generates the list of rhymes LR, and then $\mathscr{L}$ and LR are combined by Eq. (5) to update the $\mathscr{L}$ list. The list $\mathscr{L}$ with the vocabulary is then sent to the language model module, for the selection of the most coherent option. Finally, the best option is processed with FreeLing, in order to make it respect the grammatical information provided by the POS tag in the PGS (to preserve inflection). In the Fig. 2, we have marked in blue the **First step** where the semantic vocabulary list is generated to subsequently be sent to the **Second step**, the Language Model module marked in green. The pink section, the **Rhyme analysis**, is an independent section that is only executed once in the process.

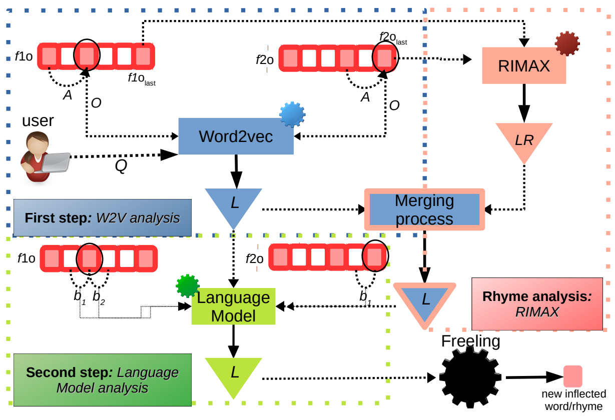

Figure 2. Second Step: Vocabulary Selection

We can illustrate an example in Spanish of our model as follows:

1. **Templates generation** (Canned Text): *Jamás he sido más [AQMS]; yo era ya un [NCMS]. / [VMI3S] el [NCMS] rápidamente hacia las [NCFP] que [VMII3P] el [NCMS].*
2. **Vocabulary selection** (Word2vec): *Jamás he sido más [afectuoso]; yo era ya un [NCMS]. / [Corría] el [sol] rápidamente hacia las [cumbres] que [anteponían] el [NCMS].*
3. **Rhymes selection** (RIMAX): *Jamás he sido más [afectuoso]; yo era ya un [**ofrecimiento**]. / [Corría] el [sol] rápidamente hacia las [cumbres] que [anteponían] el [**firmamento**].*

6. Experiments and Evaluation

In preliminary tests of our proposal, we generated and evaluated 44 pairs of rhyming sentences that were generated using PGSs created from sentences in the **MegaLite-Es** corpus. The PGSs respect the rules specified in Section 5.1. The new contexts are given by eleven different queries: *amor, odio, tristeza, alegria, sol, luna, hombre, mujer, bosque, desierto, mar* (love, hate, sadness, joy, sun, moon, man, woman, forest, desert, sea). An examples of the generated sentences are listed below, where we show the queries, the sentences in Spanish in **bold** print, and in *italic* their translation.

sadness: **El sol de mediodía encapota sobre la inexpresable niebla de mi bosque.** | *The midday sun overlays the inexpressible mist of my forest.*
Subía el sol rápidamente hacia las desolaciones que limitaban el zopilote. | *The sun was rising rapidly towards the desolations that bordered the vulture.*

Although general ATG tasks have been widely addressed by the research community, using different automatic evaluation protocols, it is difficult to implement automatic evaluation in the case of *literary text* due to the ambiguity and subjectivity involved in its interpretation and evaluation [Boden 2004]. For this reason, we have performed a manual evaluation of our experiments, asking 6 people with a graduate degree in literature to evaluate the rhymes generated by our algorithm and their semantic relations. In previous general ATG models [Moreno-Jiménez et al. 2020b, Moreno-Jiménez et al. 2020a], criteria such as coherence and grammatical composition were evaluated. Here, we asked the evaluators to indicate if they perceived a rhyme between the last words of each sentence in a pair and also to specify their perception of the semantic relation between the two rhyming words, which could be one of the following: *any relation*, *low relation*, *acceptable relation*, *good relation* and *strong relation*. We calculated the mode and median of the evaluator's feedback, obtaining a *low relation* between the two rhyming words. This was to some extent expected because, when the model looks for the second rhyming word, the semantic analysis is performed considering not only the word to rhyme, but also the general context (the *query*) and the adjacent word. For this reason, it is expected that, in some cases, the semantic relation between the two rhyming words cannot be preserved, although some relation was always perceived. For the evaluation of rhyme, we obtained encouraging results with a perception of rhymes in 61% of the pairs of sentences.

7. Conclusions and Future Work

We have proposed a model capable of generating rhyming sentences in Spanish, although with a weak semantic relation between them. This can be improved by altering the semantic analysis, when selecting the second rhyme. We have presented preliminary results showing that the model generates literary sentences that integrate semantic aspects with rhymes. Nevertheless, it must be considered that this task is still an open problem and further models, their extensions and generalizations, and experiments may confirm or improve the results that we have obtained. We expect to perform more complex and extensive evaluations, and analyse more criteria, such as coherence, by generating more sentences and recruiting more evaluators. We also plan to conduct experiments in other languages, like French or Portuguese.

Acknowledgment: We thanks CONACYT (Mexico), grant number 661101 and Agorantic (Avigon Université, France) by their financial support.

References

Agirrezabal, M., Arrieta, B., Astigarraga, A., and Hulden, M. (2013). Pos-tag based poetry generation with WordNet. In *European Workshop on NLG '13*, pages 162–166, Sofia, Bulgaria. ACL.

Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828.

Boden, M. A. (2004). *The creative mind: Myths and Mechanisms*. Routledge.

Bracewell, D., Ren, F., and Kuriowa, S. (2005). Multilingual single document keyword extraction for information retrieval. In *Proc. ICNLPK' 05*, pages 517–522, Wuhan, China. IEEE.

Clark, E., Ji, Y., and Smith, N. A. (2018). Neural text generation in stories using entity representations as context. In *Proc. NACACL-HLT '18*, volume 1, pages 2250–2260, New Orleans, Louisiana.

Kiddon, C., Zettlemoyer, L., and Choi, Y. (2016). Globally coherent text generation with neural checklist models. In *Proc. EMNLP '16*, pages 329–339, Austin, Texas. Association for Computational Linguistics.

Medina Urrea, A. (2018). *Diccionario de rimas asonantes y consonantes del español de México*. El Colegio de México, Mexico.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. In Bengio, Y. and LeCun, Y., editors, *ICLR '13*, Scottsdale, Arizona, USA. ICLR.

Mikolov, T., Yih, W.-t., and Zweig, G. (2013b). Linguistic regularities in continuous space word representations. In *NACACL-HLT '13*, pages 746–751, Atlanta, USA.

Molins, P. and Lapalme, G. (2015). JSrealB: A bilingual text realizer for web programming. In *Proc. ENLG '15*, pages 109–111, Brighton, UK. ACL.

Moreno-Jiménez, L.-G., Torres-Moreno, J.-M., and Wedemann, R. S. (2020a). Literary natural language generation with psychological traits. In *Proc. NLPIS '20, LNCS*, volume 12089, pages 193–204, Cham. Springer.

Moreno-Jiménez, L.-G., Torres-Moreno, J.-M., Wedemann, R. S., and SanJuan, E. (2020b). Generación automática de frases literarias. *Linguamática*, 12(1):15–30.

Moreno-Jiménez, L.-G. and Torres-Moreno, J.-M. (2021). Megalite: A New Spanish Literature Corpus for NLP Tasks. In David C. Wyld, D. N. E., editor, *Proc. AIAP '21*, Zurich, Switzerland.

Oliveira, H. G. (2017). A survey on intelligent poetry generation: Languages, features, techniques, reutilisation and evaluation. In *Proc. ICNLG '17*, pages 11–20.

Oliveira, H. G. and Cardoso, A. (2015). Poetry generation with PoeTryMe. In *Proc. CCR-TCM '15*, volume 7, Paris. Atlantis Thinking Machines.

Padró, L. and Stanilovsky, E. (2012). FreeLing 3.0: Towards wider multilinguality. In *Proc. of the 8th on LREC '12*, pages 2473–2479, Istanbul, Turkey.

Torres-Moreno, J.-M. (2012). Beyond stemming and lemmatization: Ultra-stemming to improve automatic text summarization. *ArXiv*, abs/1209.3126.

Urrea, A. M. and Torres-Moreno, J.-M. (2019). RIMAX: ranking semantic rhymes by calculating definition similarity. *ArXiv*, abs/1912.09558.

van Deemter, K., Theune, M., and Krahmer, E. (2005). Real versus template-based natural language generation: A false opposition? *Computational Linguistics*, 31(1):15–24.

Zhang, X. and Lapata, M. (2014). Chinese poetry generation with recurrent neural networks. In *Proc. EMNLP '14*, pages 670–680, Doha, Qatar. ACL.