Procesamiento del Lenguaje Natural, Revista nº 74, marzo de 2025, pp. 321-333 recibido 05-12-2024 revisado 31-01-2025 aceptado 12-02-2025

The tiny poet: artificial poetry generation with a constrained GPT-2 model

The tiny poet: generación de poesía artificial con un modelo GPT-2 restringido

Sergiu Stoia, L. Alfonso Ureña-López, Arturo Montejo-Ráez
Universidad de Jaén, Spain
{sstoia, laurena, amontejo}@ujaen.es

Abstract: This paper presents a GPT-2 based constrained language model trained for poetry generation in Spanish. Our proposal applies constraints to the generated sequences to satisfy rhyme and meter, by means of a backtracking process in the text generation process. For its evaluation, a Turing test has been carried out on a sample of lay population, and an evaluation of several factors on a Likert scale by experts. Despite the relative simplicity of the GPT-2 model compared to current ones, the results obtained highlight the value of constraint-based generation systems as opposed to models with a larger number of parameters and which are far more expensive to train.
Keywords: Automatic Language Generation, Constrained Language Model, Artificial Poetry.

Resumen: Este trabajo presenta un modelo del lenguaje con restricciones basado en GPT-2 entrenado para la generación de poesía en español. Nuestra propuesta aplica restricciones a las secuencias generadas para satisfacer rima y métrica, mediante un proceso de backtracking en el proceso de generación de texto. Para su evaluación se ha llevado a cabo un test de Turing sobre una muestra de población lega, y una evaluación de diversos factores sobre una escala de Likert por expertos. A pesar de la simplificidad relativa de GPT-2 frente a modelos actuales, los resultados obtenidos ponen en valor los sistemas de generación basados en restricciones frente a modelos con mayor número de parámetros y más costosos de entrenar.
Palabras clave: Generación automática del lenguaje, Modelo del lenguaje con restricciones, Poesía artificial.

1 Introduction

Within the past few years, Artificial Intelligence (AI) has been gaining presence in many fields, all driven by the use of machine learning techniques and deep neural networks used in Deep Learning.

One of the most relevant areas within AI is Natural Language Processing, an interdisciplinary area that involves both Computer Science and Linguistics, with the main objective of developing software systems designed to process human language. As a result of the advances in this field, systems that analyze, understand and generate language have been achieved.

The task of natural language generation has taken great relevance thanks to Deep Learning models, among which Transformers architecture (Vaswani et al., 2017) plays a prominent role being the cornerstone on which many of today's models are based, replacing models related to different types of architectures such as Recurrent Neural Networks.

Generative Pre-trained Transformer (GPT) models (Radford and Narasimhan, 2018) managed to outperform most of the models on the market by using large corpus of unlabeled text, followed by discriminative fine-tuning for specific tasks, which GPT3

ISSN 1135-5948 DOI 10.26342/2025-74-22

attempts to avoid with the *few-shot learning* methodology(Brown et al., 2020b), reducing data collection effort and computational costs.

The level of achievement reached by these novel models is unbeatable, outperforming previous systems in many tasks (Bang et al., 2023). But the main drawback with these models is that they operated as black-boxes, without access its internals, so parameters or other configuration aspects remain hidden.

The goal of this paper is to present an artificial poetry generation system based on the use of constrained language generation and a language model. In order to achieve this final goal, it has been necessary to address certain specific objectives:

- It is convenient to perform a process called fine-tuning to the pre-trained model for it to be capable of generating poetic texts properly.
- To develop an automatic method able to measure the meter of a given verse
- To design an algorithm that finds the rhyme between two words if it exists.
- Lastly, the final generation system will be implemented with the constraints mentioned above, this system will use the fine-tuned language model. During the generation process, the restrictions of meter and rhyme are checked, forcing a backtracking process (similar to a beam search) to opt for a different token in the sequence being generated. Figure 1 shows this process which is explained in detail later.

This paper is structured as follows: Section 2 reviews previous work related to poetry generation and constrained language generation, Section 3 mentions those materials and methods employed in the completion of this system, Section 4 describes the proposed approach, Section 5 exhibits the evaluation method, Section 6 offers concluding remarks and mentions future work.

2 Previous work

The task of artificial poetry generation is a popular task belonging to the area of automatic language generation and computational creativity(Mcgregor, Purver, and Wiggins, 2016). As a result of the evolution of language technologies and the changing poetic traditions between languages, there has been different approaches applied throughout the years (Gonçalo Oliveira, 2017).

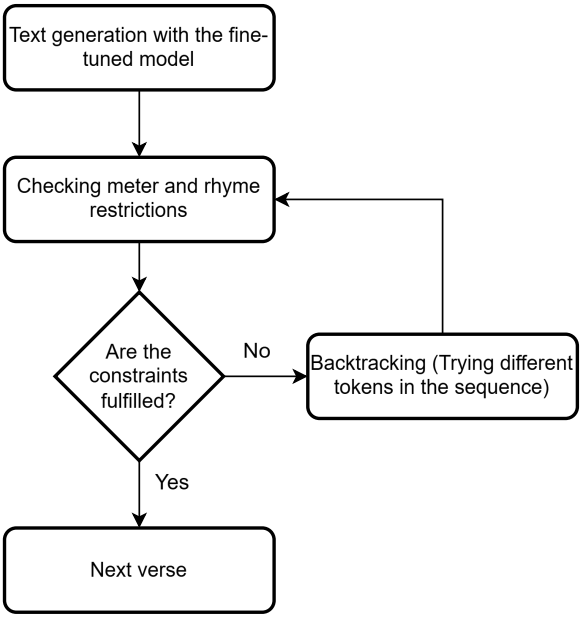

Figure 1: Flow chart showing the process of generating a new verse.

Spanish was one of the first languages that related this topic to AI (Gervás, 2001) (Díaz-Agudo, Gervás, and Gonzalez-Calero, 2002) by means of case-based reasoning techniques, this was followed by the use of other techniques such as evolutionary algorithms (Manurung, 2003) or template based approaches (Gonçalo Oliveira, 2012). Neural networks, and especially deep learning, have taken an important role on this topic through LSTM-based language models (Lau et al., 2018) and Transformers (Popescu-Belis et al., 2022). The work where the Transformer architecture was proposed (Vaswani et al., 2017), already explored beam search to improve generation, a technique well-known in language modeling.

Beam search is a heuristic search algorithm commonly used in language models for text generation. It works by maintaining a set of partial hypotheses and expanding them incrementally, keeping only the most promising ones at each step (Federico et al., 1995). Constrained beam search works by injecting desired tokens (Garneau and Lamontagne, 2023) or phrases (Shen et al., 2022) at every step of the

generation, effectively forcing the model to include certain elements in its output. It has been successfully applied to text summarization (King et al., 2022), code generation (Zhang et al., 2023) or machine translation and rewriting (Post and Vilar, 2018; Hu et al., 2019).

To the best of our knowledge, no prior work has been published applying constrained language modeling for poetry generation in Spanish. Besides, the GPT-2 model and architecture (Radford et al., 2019) has been applied in generating poetry in both English (Lo, Ariss, and Kurz, 2022) and French (Hämäläinen, Alnajjar, and Poibeau, 2022), although it has not been applied to Spanish so far.

Large language models that offer pre-trained instances on the decoder, auto-regressive side, such as GPT-3 (Brown et al., 2020a), have shown impressive performance in generating coherent and contextually relevant texts. These models have burst into the field of dialogue systems and text generation research to change it forever, due to their superior capabilities in language understanding. In fact, conversation-oriented models are becoming the most advanced, with some of the most revolutionary being the well-known ChatGPT(Zhou et al., 2023), LaMDA (from Google)(Cohen et al., 2022), LLaMA (from Meta) (Touvron et al., 2023), and OpenAI's GPT-4 (Open AI, 2023).

When we ask ChatGPT to generate a sonnet about the Spring season, the output is of high quality, maintaining rhyme. Here is the output (in Spanish):

Llega la primavera con su encanto
y trae consigo el florecer del suelo,
las aves cantan su dulce canto
y el sol luce radiante en el cielo.

Los árboles se visten de color
y el aire huele a frescura y vida,
las mariposas revolotean con amor
y la naturaleza se siente querida.

Los días se alargan y la luz se intensifica,
los corazones se llenan de alegría,
la tristeza se desvanece y la esperanza aflora.

Primavera, estación de la renovación,
de la fecundidad y la creación,
gracias por regalarnos tu bendición.

By GPT-3.5 Turbo

As we can see, the meter is not kept for what is expected for a sonnet (many of them are not hendecasyllabic) and, though we can identify the first to quatrains and the last two tercets, the rhyme is not following the expected ABBA, ABBA, CDC, DCD structure or any of the possible variants typical within sonnets. A, B, C and D represent the position of the rhyme at the verse level, so ABBA means that, in the first stanza, first and fourth verses rhyme, as for second and third verses. Therefore, even with one of the most advanced language models nowadays, the task of writing poetry has not been accomplished. Surprisingly, when asking the same for the most advanced GPT-4 model, the result is even worse:

En el tierno abrazo de la luz primaveral,
donde flores brotan y verdes enredaderas se enlazan,
una sinfonía de vida prende en la noche,
como dulces melodías del verso natural.

Desde el sueño profundo, la Tierra a su belleza llama,
cada capullo y hoja, valiente y vibrante matiz,
la lluvia suave, en tiernos besos cae,
para nutrir vida, y esperanza y alegría renovar.

El sol asciende en los cielos azules,
y calienta los corazones de seres grandes y pequeños,
un aliento de vida en cada suspiro de brisa,
un mundo maravilloso y vibrante en colores se extiende.

Oh, Primavera, que tu gracia y encanto reine siempre,
en cada flor, tu amor y luz se

sostiene.

By GPT-4

This poem, though it could be confused with a human-generated one, is failing in meter and rhyme at a larger extent compare to its predecessor, the ChatGPT model. Surpresingly, it has been found recently that OpenAI models may be preferred by many readers (Porter and Machery, 2024). But those models are expensive, yet not attending for correctness in poetry styles. This is why constrained language models can perform better when the expected generated text has to attend to very specific restrictions, like those found in poetry.

Our work demonstrates, as it major contribution, that it is possible to achieve artistically attractive yet stylistically correct verses using significantly smaller models than the current LLMs.

2.1 Constrained language generation

Nowadays, generative language models are able to produce coherent and fluent texts in a multitude of natural language processing tasks, achieving sentences that can have any shape. However, in real-world applications it is often required that the output follows a certain structure, which calls for the imposition of constraints on the obtained texts. This tasks is known as constrained language generation and it has been usually applied in code generation, so the code is valid and interpretable/compilable by a computer, as the PICARD system (Scholak, Schucher, and Bahdanau, 2021), based on the T-5 model and applying incremental parsing for ensuring constrain accomplishment.

Constrained language models have also been recently applied to generate English artificial poems (Roush et al., 2022). The method proposed, called *Gadsby*, is based on output filtering, but do not incorporate backtracking or constrain optimization as is present in our approach.

Imposing constraints on the output generated by language models helps real-world applications achieve more useful and safer results in a wide range of scenarios. Since most common models do not offer much control over the generated sequence, caused by the necessity of re-training large transformer-based models and the lack of support for imposing constraints during the generation of text, this task proves to be quite complicated (Garbacea and Mei, 2022).

To the best of our knowledge, this is the first time that a constrained language model has been proposed to generate artificial poetry in Spanish. Our approach applies an adaptation of the beam search algorithm (Wang et al., 2021) as backtracking strategy.

3 Material and methods

The resulting system uses a pre-trained model in Spanish based on GPT-2, since it is a model that adapts to the Causal Language Modeling task, a methodology that seeks to predict the next token in a sequence. This model has been pre-trained using a Spanish corpus composed of 570 GB of texts collected by the *Biblioteca Nacional de España*(Fandiño et al., 2022) and has been obtained from the Huggingface website[1]. The architecture of this model is 12-layer, 768-hidden, 12-heads, 117M parameters. The training corpus was tokenized using a byte version of Byte-Pair Encoding (BPE) (Shibata et al., 1999) used in the original GPT-2 model with a vocabulary size of 50,262 tokens.

In order to create poetic texts, it is necessary to perform a fine-tuning of the model, a process that tries to adjust the parameters of a model in such a way that it is targeted to a more specific purpose. This process has been completed with a corpus of poetry in Spanish that we have compiled [2] from the Miguel de Cervantes Digital Library[3] (Bia and Pedreño, 2001), retrieving works authored by a total of 109 of the most influential authors from 15 Spanish-speaking countries, consisting of a total of 5262 poems from which 772,746 tokens are obtained after removing punctuation marks and blank lines.

The fine-tuning process was carried out with a learning rate of 2e-05 and a batch size of 4. Two data splits were created from the corpus to apply an early stopping strategy in order to avoid over-fitting.

For fine-tuning the model and generating the poems we have used a computation

[1]https://huggingface.co/PlanTL-GOB-ES/gpt2-base-bne

[2]https://anonymous.4open.science/r/hispanic$_p$*oetry*

[3]https://www.cervantesvirtual.com/

node with four GPU NVIDIA Tesla A100 (Ampere) with 40GB RAM each. The hyper-parameters used were the default ones offered by the Transformers library implementation of the GPT-2 model of the Huggingface project.

Although this fine-tuned model is able to generate poetry-focused texts, it does not have into consideration certain rules inherent to this literary genre, such as the structure of a poem, including rhyme and meter. To meet these rules, it is necessary to customize the output of the resulting model by means of constraints related to rhyme and meter.

Our proposal generates poems starting with a configuration set by the user. This configuration is made up of two main constraints (meter and rhyme) and an initial text sequence from which the model will continue the auto-regressive generation. An example of this configuration is provided in Section 4.1.

3.1 Meter constraint

Metrics in poetry represent the construction of the poem by using a syllabic count that relates to the number of syllables in each verse. In the Spanish language, the meter of a poem is not simply the number of syllables in a line, since there are certain conditions that can modify this measurement:

- Depending on the position of the stressed syllable:
  - Last position: the final measure will be increased by 1
  - Ante-penultimate position: the final measure will be decreased by 1
- If both the last character of a word and the first character of the following word are vowels, those syllables will be merged counting as only one in the final measure, this occurrence is known as *synalepha*:

va_a *dar* ***agua_a*** *su caballo*

In order to obtain all the syllables that compose a text, we have used the *syltippy* [4] package.

[4] https://github.com/nur-ag/syltippy

3.2 Rhyme constraint

Rhyme is defined as the sound correspondence between words or word endings, especially when these are used at the end of lines in poetry. In Spanish, rhyme can occur between vowels or between combinations of vowels and consonants. This poetic device involves two exceptions that will affect the result when it comes to detecting whether two words rhyme or not:

- Exception for diphthongs: To achieve the rhyme of two words, it is possible to eliminate the weak vowel of a diphthong. For example, ac**eite** rhymes with v**ete**
- Exception for words having the ante-penultimate syllable as the stressed syllable: The syllable following the stressed syllable can be ignored. For example, c**án**ti**co** rhymes with z**anco**

4 Proposed approach

The proposed solution leverages a backtracking mechanism, which consists of a stack, where each element of the stack represents a verse of the poem to be generated, which enables the system to perform backtracking as shown in Algorithm 1. This method allows the system to return to previous steps, in the case where favorable results are not found after several attempts to generate a verse that satisfies all the constraints. This strategy ensures that the system is not trapped in an endless cycle while trying to generate a single verse.

The poem generation algorithm uses the fine-tuned language model to construct each verse, this process is reflected in the Algorithm 2. The model takes as input the previously generated verses or, if none have been generated, the initial text, resulting in a generated text to which rhyme and meter constraints will be applied in order to obtain a suitable verse, customizing the content of the generated text if required with alternatives coming from the generative model itself.

Since auto-regressive models generate distribution of probability over the entire vocabulary, the highest-scoring token is typically selected as an option for the next token for a given sequence. If accepted, that token is added to the sequence with the rest of the tokens that are part of the probability

distribution. Consequently, if the final token from a verse does not meet the constraints during validation, the system pops a new token from the stack. If all potential tokens are exhausted, the system can backtrack further, discarding the previous token and exploring alternative paths.

Algorithm 1 Poem Generation Process

poemStack will help to perform backtrack
while poem not completed **do**
  $newVerse \leftarrow createNewVerse()$
  while $newVerse$ not accomplished **do**
    $newVerse \leftarrow poemStack.pop()$
    $newVerse \leftarrow performBacktrack()$
  end while
  Adding $newVerse$ to *poemStack*
end while

Algorithm 2 Verse Generation Process

candidates $\leftarrow$ *model.generate*()
for each *candidate* in *candidates* **do**
  $metric \leftarrow computeMetric(candidate)$
  if $metric > intendedMetric$ **then**
    Adjust the verse by removing words
  else if $metric < intendedMetric$ **then**
    Complete verse using the model
  end if
  Validate the *candidate* for rhyme, style, and completeness.
  if *candidate* is valid **then**
    return *candidate*.
  else
    Use the model to predict tokens for *candidate* that meet constraints
  end if
end for
return failure in generating verse

4.1 Architecture

In the first step, the poetic structure and the initial text from which the model will start to generate the final poem will be collected from the user's input. The system is capable of generating any desired structure, adjusting the number of lines, stanzas, meter of each verse, and rhyme as follows:

6a/6b/ /7a/7b/ /5-/6a

In this representation, each number (the metric of the verse) followed by a letter (the rhyme concordance) is a verse, white spaces stand for a new stanza, while hyphen indicates that the verse lacks rhyme.

After collecting these inputs, the system will start generating the poem verse by verse, performing backtracking if necessary.

4.1.1 Verse generation

In this phase, the language model is used to generate segments of the verse until it is complete, applying the aforementioned restrictions by modifying the generated text. The model takes as input the previously generated verses or, if none have been generated, the initial text.

During this process, if the generated text exceeds the required metric, the system will remove the words contributing to the excess until the metric aligns with or falls below the expected metric for the verse.

If all the constraints are still not satisfied, then some of the last words from the text will be removed, in which case the model will be used to predict words that meet both, meter and rhyme constraints. An example of this procedure is shown below:

al ocaso se ocultara en los alrededores ✗

al ocaso se ocultara en el ocaso ✓

If a suitable verse is not found after several attempts, this generation step will be declared as failed, causing the system to take a step backwards.

4.2 Heuristics

One of the main drawbacks encountered in the generation of poetry with this approach is related to rhyme, as this constraint causes the execution of this system to take a long time.

To optimize rhyme resolution, we have implemented a dictionary based on the rhyme between words, inspired by existing dictionaries [5]. With such as dictionary, each type of rhyme found in the corpus will provide a list of all the words that rhyme with each other. An example of an entry would be:

'ao': [pato, patos, rato, manzano, ...]

The resulting system uses this resource during the generation of each verse, more specifically, when removing the last words of each generated text. The language model will

[5] https://www.cronopista.com/dict-fe/

be used to predict the next most likely tokens using a top-k decoding strategy, to which this dictionary will be applied, checking if any of the next most likely tokens exist in these dictionary entries.

5 Evaluation

After implementing the proposed approach, we achieved a system that is able to fulfill all the rhyme, meter and structure constraints when generating poetry. For evaluation purposes, we generate 21 different poems which forms are taken from random corpus texts, as it is seen in both annexes A and B, the system is able to achieve grammatically correct and quite cohesive results in some cases.

We performed a Turing test with these results, combined with 20 random poems from the same corpus used for fine-tuning. For this test, we have used Doccano, an open source text annotation tool [6]. The test consists of two different labels that define whether the poem is real or generated by the system, with a total of 20 participants, including people with varying levels of expertise in this literary genre, being a set of assessors quite diverse in educational and academic background.

In addition, a separate evaluation, with different poems, has been performed with literary and philological experts. The selection of questions for this study follows the evaluation approach of previous poetry generation systems (Shihadeh and Ackerman, 2020). For each of the 22 texts that have been evaluated, including real poems as well as poems generated by the final system, the participants were asked the following questions:

1. How typical is this poem?
2. How good is the use of language?
3. Does the text evoke mental images?
4. Does the text evoke emotions?
5. Rate the text based on how much you liked it.

Each question has been answered using a Likert scale, from 1 (Strongly disagree) to 5 (Strongly agree).

[6]https://doccano.github.io/doccano/

5.1 Results and discussion

The results obtained from the Turing test show two type of errors that can be made by participants (Figure 2).

- Mistaking a system-generated poem for a real poem (Type 1 error or false positive)
- Confusing a real poem with a poem generated by the system (Type 2 error or false negative)

We have obtained 820 labels from 20 participants, 143 of these labels (Figure 2) belong to type 1 error.

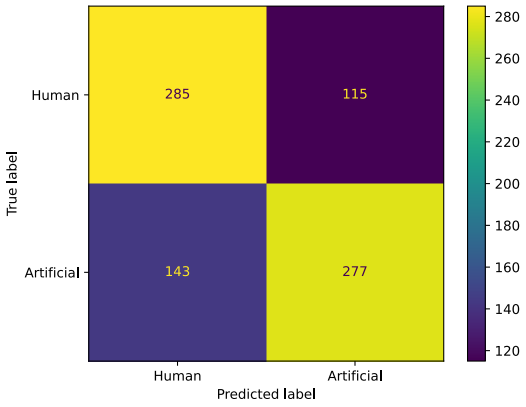

Figure 2: Confusion matrix between Human and Artificial poem annotations collected from the Turing test.

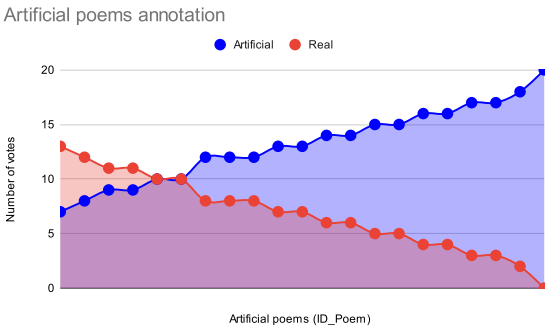

Figure 3: Results collected by artificial poems.

Out of all the real poems (positive) predicted by participants, the percentage of truly positive predictions (precision) is 66.58%. This means that users mistake artificial poems for real poems with an

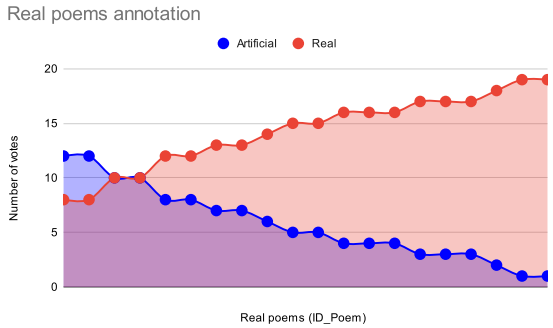

Figure 4: Results collected by real poems.

error rate of 33.42%, with fair agreement among the annotators since the correlation coefficient between participants turns out to be 0.2153 (Table 1).

Figures 3 and 4 provide a visual estimation of the evaluation results. In Figure 3 we can see that, although the majority of the annotations succeeded in determining the artificial origin of the poems, there is a relevant number of them which were considered to be written by human authors. Actually, six of them where found to be real poems by its majority of votes. The red area is the smaller one, but the purple one is also big. When contrasting this figure with Figure 4, it is clear that our annotators also have some difficulties to discriminate real poems, although the overlap in this case is smaller, but not too far from what we see when evaluating artificial poems. The conclusions that can be drawn from these two figures is that it is more difficult to recognise an artificial poem than to recognise a real one. There is less consensus in the former case than in the second one.

The main problem found with these results is the lack of cohesion between verses as it is seen in the annex B where we collected the artificial poems that have been most detected by participants.

The Likert test corroborates the observations obtained from the Turing test. As it is reflected in Figure 5 and Figure 6, the scores are lower for the system-generated poems compared to the real poems, particularly in the aspects of 'use of language' and 'evocation of mental imagery'. These results suggest that, although the system can generate poems with the correct formal structure, it still has difficulties in producing rich and evocative

Coefficient	Result
Cohen's kappa	0.2153
Pearson	0.2219

Table 1: Correlation coefficient between annotators in the Turing test

poetic language.

It is interesting to highlight that poem number 3 (Annex C) obtained high scores in the majority of the questions, even surpassing some actual poems. This particular case suggests that the system can, on certain occasions, produce poems that approach the quality of human poetry in terms of language and imagery.

The average scores reflected in Figure 6 are higher for the real poems, with large differences found in the questions 'How good is the use of language' and 'Does the text evoke mental images', which relates to the problems found in the Turing test, where we find texts lacking cohesion and correct use of language.

Overall, the results of both tests suggest that the implemented system is capable of generating artificial poems that can be mistaken for real ones, proving the effectiveness of this approach.

6 *Conclusions and future work*

In this paper, we presented an artificial poetry generation system. The proposed approach consists of a pre-trained GPT-2 model, which has been fine-tuned with a Hispanic poetry corpus, and constraints based on form, meter and rhyme. The combination of these elements result in a system that takes into account the form of the generated output, which is defined by the user, and creates text from seed words.

The heuristics proposed to speed up the search for a valid rhyme has been fruitful, reducing the time of generation of, for instance, a sonnet from 40 to 90 minutes to around 3 minutes. We are sure that more strategies can be explored to make the generation of text be done in seconds.

The combination of language models and explicit constraints proved to be useful for this task, achieving a fully functional system that generates valid poetic texts, but which may lack of cohesion and meaning. This problem is common amongst most poetic text generation systems(Popescu-Belis et al.,

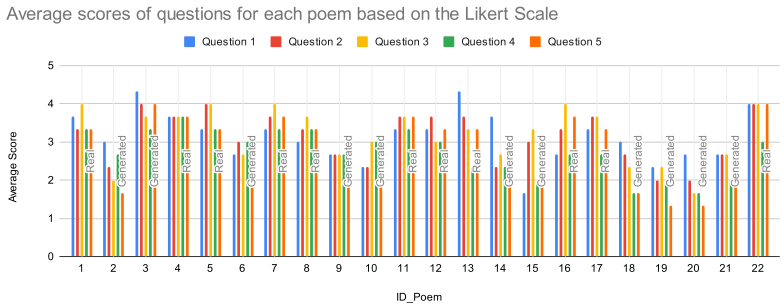

Figure 5: Average scores for each poem in Likert scale.

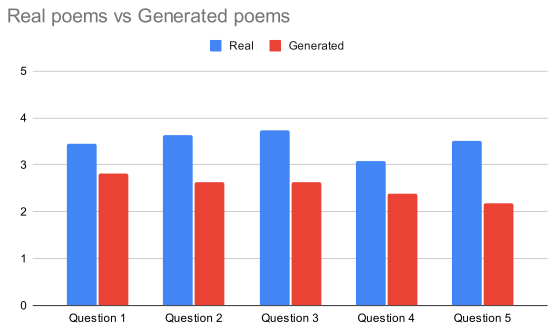

Figure 6: Average scores for each question in Likert scale.

2022), this could be solved by using larger language models such as T-5 (Raffel et al., 2020) or the more recent Alpaca (Taori et al., 2023) (a more compact version of LLaMA), both freely available, or by creating larger and more specific corpora that considers topic or emotions in poems.

In general, lay people tend to find difficulties to difference human-written poetry from machine-generated ones, which is in line with recent findings (Porter and Machery, 2024). Our main contribution is that for this task, and presumably for many others, not so large models are needed if an effective strategy of constrained generation can be defined.

Acknowledgements

This work has been partially supported by projects Big Hug (P20_00956, PAIDI 2020) and WeLee (1380939, FEDER Andalucía 2014-2020) both funded by the Andalusian Regional Government, and projects CONSENSO (PID2021-122263OB-C21), MODERATES (TED2021-130145B-I00), SocialTOX (PDC2022-133146-C21) funded by Plan Nacional I+D+i from the Spanish Government, and project PRECOM (SUBV-00016) funded by the Ministry of Consumer Affairs of the Spanish Government.

References

Bang, Y., S. Cahyawijaya, N. Lee, W. Dai, D. Su, B. Wilie, H. Lovenia, Z. Ji, T. Yu, W. Chung, Q. V. Do, Y. Xu, and P. Fung. 2023. A Multitask, Multilingual, Multimodal Evaluation of ChatGPT on Reasoning, Hallucination, and Interactivity, February. arXiv:2302.04023 [cs].

Bia, A. and A. Pedreño. 2001. The miguel de cervantes digital library: the hispanic voice on the web. *Literary and linguistic computing*, 16(2):161–177.

Brown, T., B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. 2020a. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Brown, T. B., B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. 2020b. Language models are few-shot learners.

Cohen, A. D., A. Roberts, A. Molina, A. Butryna, A. Jin, A. Kulshreshtha, B. Hutchinson, B. Zevenbergen, B. H. Aguera-Arcas, C. ching Chang, C. Cui,

C. Du, D. D. F. Adiwardana, D. Chen, D. D. Lepikhin, E. H. Chi, E. Hoffman-John, H.-T. Cheng, H. Lee, I. Krivokon, J. Qin, J. Hall, J. Fenton, J. Soraker, K. Meier-Hellstern, K. Olson, L. M. Aroyo, M. P. Bosma, M. J. Pickett, M. A. Menegali, M. Croak, M. Díaz, M. Lamm, M. Krikun, M. R. Morris, N. Shazeer, Q. V. Le, R. Bernstein, R. Rajakumar, R. Kurzweil, R. Thoppilan, S. Zheng, T. Bos, T. Duke, T. Doshi, V. Y. Zhao, V. Prabhakaran, W. Rusch, Y. Li, Y. Huang, Y. Zhou, Y. Xu, and Z. Chen. 2022. LaMDA: Language Models for Dialog Applications. In *arXiv*.

Díaz-Agudo, B., P. Gervás, and P. Gonzalez-Calero. 2002. Poetry generation in colibri, 09.

Fandiño, A. G., J. A. Estapé, M. Pàmies, J. L. Palao, J. S. Ocampo, C. P. Carrino, C. A. Oller, C. R. Penagos, A. G. Agirre, and M. Villegas. 2022. Maria: Spanish language models.

Federico, M., M. Cettolo, F. Brugnara, and G. Antoniol. 1995. Language modelling for efficient beam-search. *Computer Speech and Language*, 9(4):353–380.

Garbacea, C. and Q. Mei. 2022. Why is constrained neural language generation particularly challenging?

Garneau, N. and L. Lamontagne. 2023. Guided beam search to improve generalization in low-resource data-to-text generation. In *Proceedings of the 16th International Natural Language Generation Conference*, pages 1–14.

Gervás, P. 2001. An expert system for the composition of formal spanish poetry.

Gonçalo Oliveira, H. 2017. A survey on intelligent poetry generation: Languages, features, techniques, reutilisation and evaluation, September.

Gonçalo Oliveira, H. 2012. Poetryme: a versatile platform for poetry generation, 08.

Hu, J. E., H. Khayrallah, R. Culkin, P. Xia, T. Chen, M. Post, and B. Van Durme. 2019. Improved lexically constrained decoding for translation and monolingual rewriting. In J. Burstein, C. Doran, and T. Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 839–850, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Hämäläinen, M., K. Alnajjar, and T. Poibeau. 2022. Modern french poetry generation with roberta and gpt-2.

King, D., Z. Shen, N. Subramani, D. S. Weld, I. Beltagy, and D. Downey. 2022. Don't say what you don't know: Improving the consistency of abstractive summarization by constraining beam search. In A. Bosselut, K. Chandu, K. Dhole, V. Gangal, S. Gehrmann, Y. Jernite, J. Novikova, and L. Perez-Beltrachini, editors, *Proceedings of the 2nd Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 555–571, Abu Dhabi, United Arab Emirates (Hybrid), December. Association for Computational Linguistics.

Lau, J. H., T. Cohn, T. Baldwin, J. Brooke, and A. Hammond. 2018. Deep-speare: A joint neural model of poetic language, meter and rhyme, 07.

Lo, K.-L., R. Ariss, and P. Kurz. 2022. Gpoet-2: A gpt-2 based poem generator.

Manurung, R. 2003. An evolutionary algorithm approach to poetry generation, 01.

Mcgregor, S., M. Purver, and G. Wiggins. 2016. Process based evaluation of computer generated poetry, 01.

Open AI. 2023. GPT-4 Technical Report.

Popescu-Belis, A., À. Atrio, V. Minder, A. Xanthos, G. Luthier, S. Mattei, and A. Rodriguez. 2022. Constrained language models for interactive poem generation, June.

Porter, B. and E. Machery. 2024. Ai-generated poetry is indistinguishable from human-written poetry and is rated more favorably. *Scientific Reports*, 14(1):26133.

Post, M. and D. Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural

machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1314–1324.

Radford, A. and K. Narasimhan. 2018. Improving language understanding by generative pre-training.

Radford, A., J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Raffel, C., N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1), jan.

Roush, A., S. Basu, A. Moorthy, and D. Dubovoy. 2022. Most language models can be poets too: An AI writing assistant and constrained text generation studio. In *Proceedings of the Second Workshop on When Creative AI Meets Conversational AI*, pages 9–15, Gyeongju, Republic of Korea, October. Association for Computational Linguistics.

Scholak, T., N. Schucher, and D. Bahdanau. 2021. PICARD: Parsing incrementally for constrained auto-regressive decoding from language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9895–9901, Online and Punta Cana, Dominican Republic, November. Association for Computational Linguistics.

Shen, C., L. Cheng, L. Bing, Y. You, and L. Si. 2022. Sentbs: Sentence-level beam search for controllable summarization. *arXiv preprint arXiv:2210.14502*.

Shibata, Y., T. Kida, S. Fukamachi, M. Takeda, A. Shinohara, T. Shinohara, and S. Arikawa. 1999. Byte pair encoding: A text compression scheme that accelerates pattern matching.

Shihadeh, J. and M. Ackerman. 2020. Emily: An emily dickinson machine. In *International Conference on Innovative Computing and Cloud Computing*.

Taori, R., I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford$_alpaca$.

Touvron, H., T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample. 2023. Llama: Open and efficient foundation language models.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. 2017. Attention is all you need.

Wang, J., X. Zhang, Y. Zhou, C. Suh, and C. Rudin. 2021. There Once Was a Really Bad Poet, It Was Automated but You Didn't Know It. *Transactions of the Association for Computational Linguistics*, 9:605–620, 07.

Zhang, S., Z. Chen, Y. Shen, M. Ding, J. B. Tenenbaum, and C. Gan. 2023. Planning with large language models for code generation.

Zhou, C., Q. Li, C. Li, J. Yu, Y. Liu, G. Wang, K. Zhang, C. Ji, Q. Yan, L. He, et al. 2023. A comprehensive survey on pretrained foundation models: A history from bert to chatgpt. *arXiv preprint arXiv:2302.09419*.

A Annex 1: Samples of generated poems usually confused with human authored ones

1

bella y sencilla como solía
la vi cuando era una niña preciosa
que el sol salía por el mediodía
hermosa de frente y de corona

bella como la tarde lejana
la noche en que las estrellas se desprenden
bella de mirar tan clara mirada
como el alma de una flor inocente

a veces en la penumbra del claro
pálido reflejo de la profunda
pálida y turbia noche me trajo
con su dulce voz la dulce criatura

aquella que en los ojos dejó
en el beso de despedida abierto
en aquel beso que se perdió
cuando la luz apagó los tormentos

2
ella es la que me salvó
me curó y me sanó
ella es mi ángel de dolor
de llanto desesperado
de angustia y confusión
de desventura divina
y el rigor de rigor
de tormento interminable
que no tiene corazón
ni tiene necesidad

3
me gusta el color gris y la superficie
la claridad del blanco
el blanco de la nieve y el hielo transparente
y los dos grises que no tienen esperanza
la palabra del hielo
el sonido del agua
el crujir de los huesos del frío y del ejercicio
el chirrido de las ruedas del carro siniestro
el ruido de unos pasos en la profundidad
el temblor de una boca que nunca ha podido
pensar
el silencio esperando

la sombra de un cadáver que sobrevive
a pesar de contemplar
y de no ser contemplado en el espejo claro
el frío de su carne que se resiste silenciosa
como un niño a la muerte de sus propios
ojos terribles

4
días fríos y nublados
días de lluvia dorada
de amor de otoño y lágrimas
que no son más peligrosos
pues el hombre es un tesoro
un bien que guarda esperanzas

B Annex 2: Samples of generated poems usually detected as artificial

1
los ojos son la ventana transparencia
por arrancar
la luz es el sueño que el amanecer
su sonrisa natural

2
un grito de guerra y oscuridad
un rumor atrevido
una idea que se mueve
se agita y se desvanece más
más profunda tristeza
la pena que sentimos
el dolor que nos quema también
y en el fondo del universo
y el amor profundo
la idea maravillosa
sublime y sublime salvación
la esperanza segura todavía
y al fin el sentido
la verdad y la vida después
el alma con inquietud
con angustia y con pena la vega muy
muy lejos de aquí el cielo amarillo
azulado y brillante cubierto
y los astros brillantes pasan
las hojas secas miro
miro los montes veo también
el deseo
la tristeza
la duda la amargura aún
miro el mundo la tierra viendo
el aire y las flores todavía
miro a veces tranquilo
casi dormido y casi sin fe
y pienso que si hubiera sido acá
no lo sé ahora mismo no estoy muy
muy seguro no me siento ya pos

pero aquí estoy yo como si tuviera
como yo mismo en este profundo

3
mis manos se mueven sensualmente
sus dedos no se juntan tenía
sus labios no huelen solamente
últimos besos del fuego en el aire

ivierno y negro presente
su boca está inmóvil caída
su mano y la mía lentamente
último beso el instante siguiente

ignoto y desconocido flor
de un beso que arde desesperado
su mirada perdida bajó

últimas palabras llegaron
palabras que no salen de prisión
palabras fugaces recordando

4
el río está triste la naturaleza
la madrugada
ignota la aurora parece encendida
el sueño está oscuro el alma

C Annex 3: Artificial poem with the best score from experts

Flores de la tierra, flores del campo.
¡Qué pena tan grande! ¿por qué no vives,
florecilla mía, en mi jardín triste
donde el otoño se ha levantado.

¿Por que es así como morir al cabo?
¿Y por eso has soñado ya a tu príncipe
con su palacio azul y sus jardines
tiernos ramos blancos sobre los álamos.

¡Cómo me hubiera gustado verte,
florecida entre las rosas silvestres,
contando tus soledades contado.

¡Quién sabe si son lágrimas mías
las gotas derramadas o mis risas
ocurriendo bajo un cielo gris sin estrellas.