Recurrent neural networks poetry in portuguese

Rúben Cardoso (n.81820)[1]

[1]*Departamento de Física, Instituto Superior Técnico, Universidade de Lisboa, Lisboa, Portugal*

The present article uses recurrent neural networks with long-short term memory units to generate poetry. We tested several architectures for two very different datasets fully in portuguese: *Os Lusíadas* by Luís de Camões and most of Fernando Pessoa's poetry work, using both character one hot and word based encoding. The best results, for both datasets, were obtained using networks with 2 layers and 750 units per layer, trained with character based encoded data. To achieve a balance between under and overfitting, such networks should be trained for 10 to 20 epochs.

I. INTRODUCTION

Art is often regarded as the last stronghold of humanity, the last boundary between mankind and machines. However, the development of more effective artificial intelligence techniques has contributed to the blurring of this line; specifically, the increase in computational capacity has allowed the application of deep neural networks to complex problems, such as the generation of poetry.

The study of this problem is extremely relevant because it not only contributes to expand the tasks that computers can do, but it also helps developing useful tools for natural language processing, making human-computer interactions easier.

The attempts to generate poetry with computers started in the 1960s with very simple models based on the interchange of lines from a big set of poems. Then, in the 1980s, more complex models appeared: using the structure of a poem and filling it with vocabulary from a different given set of poems.[5] It wasn't until the 21st century that Artificial Intelligence and Neural Networks started to be used; these methods enabled much better results and opened the door to new levels of insight into the text, such as context and relations between words.

The first big leap was the use of Recurrent Neural Networks (RNN) with character-based encoding, which allowed the generation of haikus and limericks with correct forms and acceptable syntax that lacked, however, appropriate relations between verses.[2] Then, with the further development of RNNs, appeared the Long Short-term memory recurrent neural networks (LSTM); these can more easily "remember"older information and reduce the disparity of ideas between different verses. [4] LSTMs have been extensively used, both with character and word based encoding, for example, to generate poetry based on Shakespeare sonnets.[7]

However, although these LSTM models have been broadly used to generate poetry in english, they haven't yet been thoroughly applied to the rich and varied portuguese poetry.

In this article, we train models for two very different types of poems: first, epic poetry using *Os Lusíadas* by Luís Vaz de Camões; and then, modernist poetry with a very considerable part of Fernando Pessoa's poetry work. These texts are either character or word encoded, and the results of this process are used to train neural networks with several different architectures. The generated poems will be analysed considering: form, the length and number of lines, rhythms and repetitions; syntax, the arrangement and ordering of words to create well-formed and consistent sentences; and verse and stanza coherency, the topics and ideas discussed are coherent with each other and the evolution of ideas makes sense.

The results obtained will serve as a benchmark for further hyper-parameter optimisation and will make clear the differences or similarities in handling and training that should be taken into account when using such distinct types of poetry.

II. METHODS

We first started by collecting the set of poems required to train the desired LSTM networks. For the case of *Os Lusíadas*, the process was simple: the online free ebooks project Gutenberg provides a copy of the full book in plain text (UTF-8 encoding), the final data was obtained after some manual preprocessing in which the text was cleaned of initial and final remarks and chapter separators.[3] The desired Pessoa's poetry work, due to its enormous variety, isn't compiled into an organised edition, thus, we developed a web scraping script to fulfil this task. This script acted on the *Arquivo Pessoa* website and downloaded the HTML code of all the available works; next, by analysing the page structure, it automatically identified and extracted the poetry works, while ignoring the remaining. We applied a final step of manual processing to remove the poems in english, ensuring that the data was totally in portuguese.[6]

Then, we developed a Python class, TextMapping, to encode the text data into an appropriate form of input for the neural network and to decode the output into regular text. Before encoding, all the raw text was converted to lower case and characters with accents were converted to their simple ASCII form.

We used two forms of encoding: character and word based encoding. For character based, the set of unique characters existent in the text was determined, including letters, numbers and punctuation marks, and an integer number is assigned for each one of the characters. The encoding was then done by replacing each character in the text by its respective number. This encoding would be enough to be accepted by the neural network as input, given that it's in a fully numerical form, but it would be highly inefficient because the data distribution is not appropriate; it would result in a very poor network convergence. Furthermore, simple numerical encoding assumes an ordinal relation between the characters which would be perceived by the model, this assumption is completely invalid and could harm the results. Thus, we performed an extra step of one hot encoding: every number was converted into a vector with length equal to the total number of characters, for this vectors all its elements are 0.0, except for the element whose position corresponds to the

desired number, which is set to 1.0.

For word based encoding, first we tokenized all punctuation marks, which means these were made into a word; for example, a comma, ',' became "‖comma‖". Then, we determined the set of unique words existent in the text and an integer number was assigned for each one of the words. As before, this encoding form is inappropriate for good performance of the network, however, for word based encoding, full one hot encoding is not an option because each number would be replaced by a vector with thousands of entries resulting from the great amount of different words in the text. The space required to accommodate such data would be enormous and, therefore, it is not feasible. As an alternative, for the data attributes, each of the assigned numbers was divided by the total number of unique words found, then it was doubled and subtracted the value 1.0. This process centres the data around a zero value and scales it to an interval $[-1, 1]$, optimising the data for the network and making convergence easier. For the labels, we used one hot encoding, this was possible because there is only one label per instance and the data was provided to the model in batches of limited size that could fit into the RAM.

Given the very considerable size of the data, we exhaustively optimised both of these methods to make the conversions as fast as possible. After encoding the texts, the data was used to train LSTM recurrent neural networks.

Artificial and convolutional neural networks function on an episodic basis, being trained and predicting based solely on the current inputs. Recurrent neural networks, on the other side, have a great advantage: these function on a sequential basis being sensitive to context and previous information, the existence of loops in the network allows information to persist. When an input is given to a certain unit, an output is, as usual, returned, but in RNNs, besides that, inside the unit a memory variable is updated and its information is used for the classification of the next input. This means that the data used for training must be a sequence of inputs, with a given length, that are successively fed to the network and only the last prediction should be compared to the single final label correspondent to that sequence. This is explicit in figure 1, the sequence is composed of $t+1$ inputs, $[x_0, x_1, ..., x_t]$, and only the final output h_t should be compared with the label.

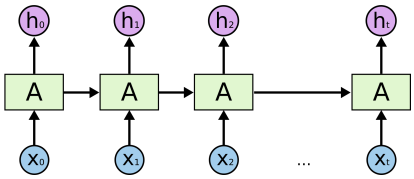

Figura 1: Temporal evolution of RNN. Horizontal arrows represent memory and x_i represent the successive elements of the sequence.

However, in practice, RNNs show difficulties in learning long-term dependencies, performing well only for problems which rely on short-term relations. To address this issue, we used long short-term memory units. These have the same functioning principle as typical RNNs but instead of keeping one memory variable, they keep two: one very slightly affected by each one of the inputs and recent past memories, for long-term dependencies; and another, that is greatly affected by this factors, for short-term dependencies.

For training purposes, we assumed the current problem as a supervised learning problem: being l the sequence size, for element n (character or word, in appropriate cases), the sequence used includes elements from n to $n+l$, and the label will be element $n+l+1$.[2] For each element in the text, a sequence is obtained. We used sequences with size 30 for word based encoding and size 100 for character based.

Camões and Pessoas' original poetry show profound differences. Taking into account the use of archaic portuguese by the first, which includes a great amount of word contractions such as "C'um"or "d'ouro", we decided to train this dataset with character based encoding. For Pessoas' poems, we first used word based encoding, to allow the network to focus more on syntax and less on word construction, and then character based encoding to compare the results.

We implemented the LSTM networks using the Python module Keras with a TensorFlowGPU backend based on CUDA, all the models were trained on a GPU NVIDIA GTX960M. Given that Keras provides an implementation of LSTM units optimised for CUDA, CudNNLSTM, we used it and were able to greatly reduce training times.

As for the networks architectures experimented, based on previously referred articles, we tested two and three layers architectures using either 450 or 750 units per layer.[2, 4, 7] For all the networks, we used a dropout rate of 20% in order to prevent overfitting. The final layer always has a number of units equal to the number of possible classes and uses a softmax activation function to allow this last layer to return the various class probabilities. Training was performed with adam optimizer and a batch size of 125. Finally, given that the problem is a non-binary classification, we selected categorical cross-entropy as the loss function.

III. RESULTS

Firstly, we tried different network architectures for Fernando Pessoa's poetry using word based encoding and the following seed: "resplandecem a distancia/ num passado impossivel de se ver/ com os olhos da fe ou os da ansia;/ lembramos nevoas, sonhos a esquecer.". The results after 40 and 90 epochs of training on a 2 layer network with 450 units per layer can be seen on poem 1 and poem 2, respectively.

Poem 1

e a vida,
a,

.
..... tremo tremo.....
tremo tremo,..... tremo.....
(...)
tremo,... sendo, fazendo o dia.

Poem 2

e a alma, e a queda,
e a hora e a ignorancia.
ele a vida,
e a realidade

a manha,

que a vida
iria beber,
tranquilos a morte

Then, maintaining the 2 layers, we trained the model using 750 units per layer. An example poem obtained after 90 epochs is shown in poem 3.

Poem 3

que crime outrora feito, que pecado
nos impos esta esteril provacao
e em ser que me vao

versos versos versos,
.. tantos...
nao vida sem eu!
deve o comboio vasconcellos de um,
e a presenca te, ou

e dor o fica, que te o meu beber.

The training of a 3 layered network with 450 units per layer did not converge.

For the case of *Os Lusíadas*, we used character based encoding and the results were obtained from the seed: "nde o governo esta da humana gente,/ se ajuntam em concilio glorioso/ sobre as cousas futuras do orien". With 2 layer and 450 units per layer, results are shown for 5 (poem 4), 70 (poem 5) and 100 (poem 6) epochs.

Poem 4

que a vista e de alta e de alta e fria,
que a vista e de alta e de alta e fria,
(...)
que a vista e de alta e de alta e fria,

Poem 5

que ele sojugara por verdadeiro.

por ele edade a forca desta gente,
se entre os deuses em pe, semple aconselha,
mas a seguir cantar do fado inimigo
por meus africa para a fatra e trante.

estava o sol que em tudo so se estende
pela aurora, sabida ja deixava,
com estas novas torna a patria cara,

Poem 6

pisando o cristalino ceu formoso,
vem todo o menos duro, que mal antes,
por que me deixas, mar a lanca e espada
de quatrocentos mouros se apartou;

As before, maintaining the use of 2 layers but increasing the number of units to 750, we obtained the poems 7 and 8, respectively after 10 and 20 epochs.

Poem 7

este como fosse alto e subtil fumo,
que a mais o mar a terra lhe contava.

e com verdade o estreito e nobre e gente,
que a mais o rei se alarga, e nao consente
de ser seu contrario de ampelusa
e de mais certas e de amores,

de quem foram contra o monte ardente,
que a mais o rei se aparto o ceu

Poem 8

entre a gente beligera vitoria;
mas nunca poder, feroz muito pelo
do mar, que do seu reino tao remoto
se faz clara a bela nobre espanha:
poder tamanho junto nao sofrendo,
estava a terra e a morte ali correndo.

e se a terra toda afonso e rico,
os deuses da carmania e nada fina,

The 3 layered network with 450 units was, for this dataset, able to converge. An example result after 70 epochs can be found in poem 9.

Poem 9

e porque estas palavras lhe apresenta
em terras grande, e a terra que se atrava
de armas, e na parte a forte armada tanta.

e se tanto tem do olimpo se parte
aqui a frota co'o descobrido e estado,
e da terra algum subtil e assentado,
e nao se achar a terra toda ardente,

Taking into account the clearly better performance of character based encoding, we trained a 2 layered network with 450 units per layer on Pessoas' poetry with this type of encoding. The result after 80 epochs can be found on poem 10.

Poem 10

que formas de tudo quanto vive sem passar?
que sombra e a vida que ha em mim e diverso...
porque e que ele esta tudo era a alma e verde
e mal tocado na rua,
e se passavas como a palavra de mim,
nao ha a casa do mundo, nao de qualquer maneira
e acaba com um sentido que me desle

Finally, maintaining two layers and using 750 units per layer, we obtained poems 11 and 12, after 10 and 20 epochs, respectively.

Poem 11

que eu sou e o que e que estou a esquecer.
a verdade e o que sou e o que e.
nao sei se poder ser sentir e sentir
e eu sentir e a minha alma e a minha vida.
tudo e misterio e o meu sentir
entre o que sou e o que espera: e e que estou.
feliz a minha alma e a terra,
dorme,

Poem 12

que tive qualquer coisa que nao e nada,
e que, porque a luz do sol vale mais que o mundo e mundo?
se eu nao tivesse a caridade...
meu deus! e esta tudo certo.
esta especie de ligacao da vida.

IV. DISCUSSION AND CONCLUSION

The results yielded by the application of a 2 layered network with 450 units per layer to Pessoa's poetry work show that this architecture is too simple to allow the proper modelling of the data. In poem 1, the high level of repetition and lack of syntax and coherency show clear undertraining. Further training, as seen in poem 2, allows for a much better use of punctuation and minimalist but almost natural form; it's also interesting to note the existence of rhythm achieved through appropriate use of repetitions.

The use of a more complex model, with 2 layers and 750 units per layers, resulted in larger verses that try to mimic natural language by using more complex syntaxes. In poem 3, the first three verses are exactly what was originally written after the provided seed. This is clear evidence of overfitting and it's not desired behaviour: the network should learn the data maintaining the ability to generalise, it should not memorise it. As for the simpler architecture, none of the present poems show verse or stanza coherency, this is likely a consequence of using word based encoding: given that data size is ≈ 200000 words and there are only ≈ 17000 unique words, this means that, assuming same word frequency, each word appears only around 10 times in the dataset. Furthermore, considering that some words have very high frequencies and other might, for example, only appear once, the model will focus on the very frequent words and won't be able to correctly learn the appropriate context for the less frequent words.

For the case of *Os Lusíadas*, the use of a 2 layered network with 450 units returned results better than any of the previous cases. Poem 4, with only 5 epochs of training, is solely composed by the repetition of a single verse which itself is also very repetitive, but, nonetheless, considering that the model is being trained on a character basis, the fundamental verse elements can already be found: words separated by spaces, correct number of characters by verse and use of punctuation marks. More importantly, it enables the correct usage and spelling of words and simple sentence syntax. Training up to 70 epochs, poem 5, allows the generation of a poem with plausible form (plausible length and number of lines), correct syntax and some level of coherency both within the verses and within the stanza. There is, however, some overfitting on the final stanza. Poem 6 is a complete copy of the original work, it indicates that character based encoding is very prone for overfitting, thus, the number of training epoch should be kept low.

Then, as done before, a network with 2 layers and 750 units per layer was used. This architecture yielded the best results for Camões' data: both poems 7 and 8 show very good form; their syntax is also considerably natural given the usual syntax used for epic poetry and there is some coherency of ideas along the stanzas. However, the most relevant property found is rhyme, there is a clear attempt to create rhyme and finish distinct verses with the same final syllable.

The use of a more complex network, poem 9, with 3 layers and 450 units per layer, returned results similar to the previous architecture. Rhyme can also be found but not as frequently as before.

Finally, a selected part of Pessoa's poetry was trained with character based encoding. This method yielded results largely better than the word based encoding. As explicit in poem 10, the simple model was able to achieve a suitable form and a syntax that, although far from natural, is much better than the achieved for word based encoding: some of the verses already show correct syntax. The use of 750 units per layer further improved the results: poems 11 and 12 show correct form and, considering the type of writing used in poetry, a very "poetic"syntax; these also show some coherency within the verses and the stanzas, generating sentences that make sense and carry an idea and maintaining a general topic throughout the stanza.

To sum up, character based encoding is clearly better than word based one, this is evident for both datasets and specially obvious for Pessoas' poetry which was trained with both encodings. Furthermore, also for both datasets, the best architecture is a 2 layered network with 750 units per layer, the use of 450 units results in a model too simple to obtain good results. Given its large number of units, overfitting is a considerable problem, thus, the number of training epochs should be between 10 and 20: a delicate balance between underfitting and overfitting. The use of 3 layered networks makes convergence harder to achieve and, when this is achieved, showed results very similar to the 2 layers with 750 units.

To further improve the results obtained, several paths can be taken, including characters with accents, testing bigger architectures with more units and more layers, increasing the fine tuning of hyper-parameters by testing a bigger variety of models (very computationally expensive) or improving the quality of Pessoa's poetry dataset by making it more consistent, both in form and vocabulary used.

Acknowledgments

We thank Project Gutenberg and *Arquivo Pessoa* for making the used datasets available and allowing their free use and distribution.[1, 6]

[1] Gutenberg project. URL `http://www.gutenberg.org`.

[2] S. Ballas. Generating poetry with poetrnn, 2015. URL `http://sballas8.github.io/2015/08/11/Poet-RNN.html`. Accessed 5 November 2018.

[3] L. V. de Camões. Os lusíadas. URL `http://www.gutenberg.org/cache/epub/3333/pg3333.txt`. Accessed 12 November 2018.

[4] T. Kiis and M. Kängsepp. Generating poetry using neural networks. *Institute of Computer Science, University of Tartu*.

[5] H. Oliveira. A survey on intelligent poetry generation: Languages, features, techniques, reutilisation and evaluation. *Association for Computational Linguistics Anthology*, 2017.

[6] F. Pessoa. Obra aberta. URL `http://arquivopessoa.net/`. Accessed 12 November 2018.

[7] S. Xie, R. Rastogi, and M. Chang. Deep poetry: Word-level and character-level language models for shakespearean sonnet generation.