Generating song lyrics using recurrent neural networks

Project for the course *Neural Networks* at Faculty of Electrical Engineering and Computing, University of Zagreb

Ena Car
Ena.Car@fer.hr

Pavao Duževič
Pavao.Duzevic@fer.hr

Damjan Grubelić
Damjan.Grubelic@fer.hr

Josipa Kaselj
Josipa.Kaselj@fer.hr

Magdalena Šimunec
Magdalena.Simunec@fer.hr

Eva Šmuc
Eva.Smuc@fer.hr

***Abstract*—This document is a project report for the Neural Networks course at Faculty of Electrical Engineering and Computing in Zagreb. It describes an implementation of a recursive neural network which is used for generating song lyrics based on given set of song lyrics.**

***Index Terms*—RNN, neural networks, generating song lyrics**

I. Introduction

Writing meaningful lyrics has always been a challenge even to the best of songwriters. Since more and more jobs are being automated, especially now with rise of popularity of various machine learning methods, there is also an initiative aimed at computers writing songs. The goal is to feed a model with lyrics of existing songs and then to use it for generating new lyrics. Those generated song lyrics aren't supposed to be word-for-word equal to those used for training the model. Still, it wouldn't be sensible having it generate random words, either. The goal is having the model memorize the lyrics of the songs it is trained on (motifs) and then combine them *sensibly*, i.e. having the model memorize "relationships" between the words through their relative order. In this project, it is being done using recursive neural networks (RNN), which are able to take in account previous outputs when generating new one, thus adding temporality. The models have been trained on several subsets of the "150K Lyrics Labeled with Spotify Valence" dataset from *kaggle.com*.

II. Existing solutions and brief literature overview

A. Automatic Rap Lyrics Generation

Variation of RNN called LSTM architecture creates a better language model than a regular RNN. [1] According to Potash, Romanov and Rumshisky, Long Short-Term Memory (LSTM) language model produces better lyrics than a baseline model. They attempted to piece together lyrics for a specific artist, but their model was limited in generating lyrics for a genre. As a result of training their model with sets of lyrics, they noticed corresponding rhyming words.

Another example of A Rap Lyrics Generator was developed by Nguyen and Sa [2]. They used database of approximately 40000 existing rap lyrics. After failure to get profound results when using linear-interpolated trigram model approach, they shifted to a quadgram model. Also, they succeeded to generate sentences that rhyme with each other. At the end, generator worked decently, but the content of the lyrics did not relate to a specific theme.

B. Automatic Generation of Poems

Wishful Automatic Spanish Poet was the first generating program for poems which used artificial intelligence and natural language generation techniques together. The whole system of WASP is based on a forward reasoning ruled-based system. Users were asked for inputs which were then used as seeds and results recieved were not very efficient in generating lyrics [3].

C. Rhyme Detection

Hirjee and Brown have developed a probabilistic model and in addition to it they have built up a rhyme detection tool based on that model [4] [5]. Tool analyzes phoneme patterns in words and model is trained on a set of lyrics that were manually annotated for rhyming words.

D. Classification of Lyrics

Mayer, Neumayer and Rauber used rhyme and style features to classify and process lyrics [6]. They followed the fact that a rhyme is two words that when spelt sound similar and generally used it for words at the end of verses.

E. Natural Language Processing and Lyrics Generation

With basic natural language processing tools, song lyrics can be analysed by using a Naive Bayes classifier. Mahedero, Cano and Martinez used that to identify languages. Also, they used it for classification based on themes and to search for similarities between them. The languages which were used for the experiment: English, Spanish, German, French and Italian. Given results were approximately 94% accurate. The conclusion they came to is that the identification of languages was easier task compared to the others.

III. Solutions implemented by the project team

Since generating the lyrics requires memorizing word ordering so that lyrics make sense, RNNs have been used, as that class of neural networks is better for memorizing sequences. Sequences used as RNN inputs can be words as well as characters and based on those two approaches, two types of models have been implemented — word-based and character-based — to generate lyrics based on the same dataset.

A. Choosing the right dataset

The lyrics needed for training such model can be scraped using the webscrapers on lyrics websites, downloaded using a Python API like *lyricsgenius* or downloaded as a readily available dataset. With the latter being the simplest method of the three, yet equally suitable, the project team opted for the "150K Lyrics Labeled with Spotify Valence" dataset because of its exhaustiveness which provides a simple way to extract almost any arists' lyrics and offers diversity of songs and genres. Since it is too big to be processed on the project team's machines, only the first 200 lyrics with more than 60% English words in them have been used for training the most of the models.

B. Preprocessing the dataset

Since the lyrics texts often contain additional info that is not part of actual song lyrics (like the chorus tag or the name of the artist performing the next verses) and since such tags are most commonly enclosed in square brackets, everything inside square brackets has been removed from the lyrics to reduce cluttering with info that may not be correct in generated lyrics and would interfere with memorizing relationships of words and motiffs.

In the **word-based approach**, however, similar interference was made in order to memorize line endings. The "word" *endl* has been added to the end of each line. That is supposed to help with rhyming and breaking lines of generated songs in meaningful places.

After adding the *endl*, each word from the dataset has been stripped of punctuation and converted to lowercase. Each word has been mapped to a unique number that would be used as network input.

Those numbers (tokenized words) have been grouped into sequences of **up to** n lines, i.e. verses. The sequences have been padded to the length of the longest sequence with zeros. The last number in each sequence would later be expected output for the input — the beginning of the sequence.

Similarly, in the **character-based approach**, the characters, instead of words, were then converted to numbers. There already was a newline character so line breaks were already covered. The converted characters have then been grouped into **exactly** m elements long sequences which were grouped into batches for training. The last number in each sequence would later be expected output for the input — the beginning of the sequence.

C. Building a model

D. Training the model

E. Generating the lyrics

F. Evaluation of the generated lyrics

IV. Project results

V. Comparison of results

By: architecture, epochs, artist and approach (in VI-C0a)

A. Conclusion

What was done, what could have been tried etc.

VI. Team member assignments and tentative time schedule

A. Team member assignments

- **Ena Car:**
- **Pavao Duževič:**
- **Damjan Grubelič:**
- **Josipa Kaselj:**
- **Magdalena Šimunec:**
- **Eva Šmuc:**

B. Tentative time schedule

C. Figures and Tables

a) Positioning Figures and Tables: Place figures and tables at the top and bottom of columns. Avoid placing them in the middle of columns. Large figures and tables may span across both columns. Figure captions should be below the figures; table heads should appear above the tables. Insert figures and tables after they are cited in the text. Use the abbreviation "Fig. 1", even at the beginning of a sentence.

TABLE I
Table Type Styles

Table Head	**Table Column Head**		
	Table column subhead	***Subhead***	***Subhead***
copy	More table copy[a]		

[a]Sample of a Table footnote.

Fig. 1. Example of a figure caption.

Figure Labels: Use 8 point Times New Roman for Figure labels. Use words rather than symbols or abbreviations when writing Figure axis labels to avoid confusing the reader. As an example, write the quantity "Magnetization", or "Magnetization, M", not just "M". If including units in the label, present them within parentheses. Do not label axes only with units. In the example, write "Magnetization (A/m)" or "Magnetization

{A[m(1)]}", not just "A/m". Do not label axes with a ratio of quantities and units. For example, write "Temperature (K)", not "Temperature/K".

Appendix

References

Please number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use "Ref. [3]" or "reference [3]" except at the beginning of a sentence: "Reference [3] was the first . . ."

Number footnotes separately in superscripts. Place the actual footnote at the bottom of the column in which it was cited. Do not put footnotes in the abstract or reference list. Use letters for table footnotes.

Unless there are six authors or more give all authors' names; do not use "et al.". Papers that have not been published, even if they have been submitted for publication, should be cited as "unpublished" [4]. Papers that have been accepted for publication should be cited as "in press" [5]. Capitalize only the first word in a paper title, except for proper nouns and element symbols.

For papers published in translation journals, please give the English citation first, followed by the original foreign-language citation [6].

References

[1] P. Potash, A. Romanov, and A. Rumshisky, GhostWriter: Using an LSTM for Automatic Rap Lyric Generation. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015.

[2] H. Nguyen, and B. Sa, Rap Lyrics Generator, unpublished, 2009.

[3] H. Manurung, G. Ritchie, and H. Thompson, Towards a Computational Model of Poetry Generation. In: *Proceedings of AISB Symposium on Creative and Cultural Aspects and Applications of AI and Cognitive Science*, pp. 79-86, April 2000.

[4] H. Hirjee, and D. Brown, Using automated rhyme detection to characterize rhyming style in rap music, 2010.

[5] H. Hirjee, and D. Brown, Rhyme analyzer: An analysis tool for rap lyrics. In: *Proceedings of the 11th International Society for Music Information Retrieval Conference*, 2010.

[6] R. Mayer, R. Neumayer, and A. Rauber, Rhyme and Style Features for Musical Genre Classification by Song Lyrics. In: *ISMIR 2008, 9th International Retrieval, 14-18 September, 2008, Drexel University, Philadelphia, PA, USA*.

[7] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

[8] I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange anisotropy," in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.

[9] K. Elissa, "Title of paper if known," unpublished.

[10] R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[11] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].