G-Rap: Interactive text synthesis using recurrent neural network suggestions

Udo Schlegel, Eren Cakmak, Juri Buchmüller and Daniel A. Keim

University of Konstanz - Data Analysis and Visualization Group
Konstanz, Germany

Abstract. Finding the best neural network configuration for a given goal can be challenging, especially when it is not possible to assess the output quality of a network automatically. We present G-Rap, an interactive interface based on Visual Analytics principles for comparing outputs of multiple RNNs for the same training data. G-Rap enables an iterative result generation process that allows a user to evaluate the outputs with contextual statistics.

1 Introduction

There are two common problems when working with neural networks: First, finding a fitting network architecture for a certain task which is often even for experts an iterative try-and-error process. Further, evaluating the quality of generated results with a semantic component is challenging, e.g. in images or text data. While some quality measures, for example, sharpness in images [7] or orthography in textual output [6], can be computed, it is hard to automatically judge how realistic an image is or if a text is semantically correct.

Therefore, an expert has to evaluate and find suitable neural network architectures for certain tasks. To aid experts in this practice, a set of requirements can be derived: A user should be able to *compare* and *select* the best result from multiple networks. Through an *iterative* process, the user should be able to concatenate and generate results progressively. By doing so, the user should *learn* about the overall performance of the examined networks.

Led by these principles, we developed G-Rap, an interactive, visual interface to improve the design process of recurrent neural networks (RNN) with Long-Short-Term-Memory [4] layers by allowing experts to iteratively choose the best results from different RNN architectures. G-Rap is built after Visual Analytics [5] principles and offers a feedback loop to use the expert's decisions to improve the results. Contextual quality measures assist the user in the decision process. Logging the user's decisions reveals long-term statistics of RNN selection preferences, which constitutes a quality indicator for RNNs.

In the remainder of this work, we discuss related approaches concerning the interactive text synthesis without and with neural networks. Subsequently, we explain the G-Rap concept in detail at the example of text synthesis for rap lyrics. Finally, in a discussion of our approach, we will highlight possible application fields and illustrate future work.

Konstanzer Online-Publikations-System (KOPS)
URL: http://nbn-resolving.de/urn:nbn:de:bsz:352-2-xeqgxbhdkp799

2 Related Work

The issue of computational poetry generation is a part of computational creativity, natural language processing, and artificial intelligence. Various systems and techniques were developed in recent years for poetry([10],[6]). *DopeLearning* [8] generates rap lyrics by combining existing lines from rap songs. The user is able to create rap songs by inserting a set of words and let the system suggest new lines. *Co-PoeTryMe* [11] is an interactive poetry generation tool that allows the user to insert lines and modify generated lines to create poetry with a certain surprise factor. The LyriSys [16] system supports the generation and modification of poetry by suggesting new lines using a provided topic or verse.

Recently, neural networks and their success in natural language processing have made them an alternative to the general template oriented style of poetry generation. Zhang and Lapata [19] demonstrated that RNNs can produce with a set of keywords Chinese poetry. Wang et al. [14] use a sequence-to-sequence model [1] with user provided keywords to generate Chinese poetry. Further, Wang et al. [15] proposed a planning-based RNN. The system constructs an outline of the poem based on keywords and generates the poem with the previously generated outline. Ghazvininejad et al. [2] proposed the *Hafez* system which generates a poem with a particular topic using RNNs. The system uses keywords to find rhyming words which are used to produce the poem line by line. Potash et al. [13] introduce a system to generate RAP lyrics for artists by using the artist's lyrics as a training ground and try to mirror the style of that artist with their generated lyrics.

Most of these systems using RNNs do not involve the human in the generation process. For instance, Yan et al. [17] introduced *i, Poet*, an iterative refinement process to generate Chinese poems based on Yan et al. [18] and Zhang and Lapata [19] with human interaction. A proposed updated *Hafez* [3] interactively generates poems using user provided topics.

Most systems evaluate poetry by using scores or human evaluation. For example, the evaluation bilingual evaluation understudy (BLEU) [12] or the rhyme density [8]. Calculated scores cannot judge creativity or semantic correctness. Human evaluation surveys criteria such as grammaticality, meaningfulness and poeticness [9], but depend heavily on the literary knowledge of experts.

To our best knowledge, nearly all named systems are restricted to certain input parameters such as keywords or topics. Some systems view the problem from an information retrieval perspective finding and combining the most fitting lines or just mirroring poetic styles. Further, Chinese poetry has a strict structure and methods to estimate a score for the generated poetry.

Our work focuses on the comparison of general RNN networks for text synthesis. G-Rap offers methods to explore and evaluate multiple RNNs for text synthesis: By supporting the user in finding the best possible RNN and combining outputs from multiple RNNs for certain tasks, G-Rap enables a simple and more productive way of synthesizing poetic text.

3 Concept & Application

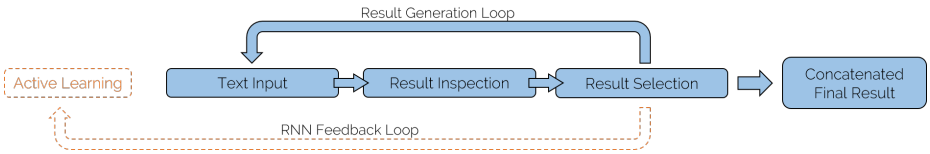

Fig. 1: The G-Rap workflow. After providing input, the user reviews and selects a result from the generated outputs and iterates the process as necessary. In future work, we intend to feedback the user-chosen results into an active learning process.

Having argued for the need of a human expert to operate the system and evaluate the results, we chose to develop G-Rap according to Visual Analytics principles [5]. Since our approach can be vaguely considered a "manual ensemble learning", the VA process helps us to integrate the machine learning side with the advantages of human cognition in a field where the results cannot be judged fully automatically.

Fig. 1 shows our conceptual approach to the issue: First, the user provides an input or the system generates a random input which is processed by all networks. Afterwards, the results of all networks are presented. Additionally, basic quality measures are presented for each result that the expert can use to contextualize the results. The chosen quality measures should assist the user in the selection process by providing meaningful additional information. For text input, such measures could be linguistic features that can be derived automatically. Also, statistics derived from previous selections help the user to identify, which network performs better for a given input.

3.1 The G-Rap System

Rules like basic vocabulary or grammar checking can help to distinguish meaningful text from incomprehensible gibberish. Yet, since language is ambiguous and interpretable, such rules can only hint at the usefulness of a text. In particular, lyrics can hardly be evaluated by automatic text evaluation frameworks. Lyrics, especially rap lyrics, tend to contain colloquial expressions to establish a flow and rhyme scheme throughout a song. Moreover, street slang consists of terms and phrases with ambiguous meaning. Thus, automatic evaluation using standard dictionaries can be misleading.

With G-Rap, we leverage the ability of human experts to evaluate grammaticality, meaningfulness, and poeticness [9] and to contextualize slang expressions, helping him to identify RNN architectures that perform better than others for specific text synthesis tasks. This is supported by providing interactive feedback options to select the best performing RNN.

The models used are RNN architectures using LSTM layers, generally used to

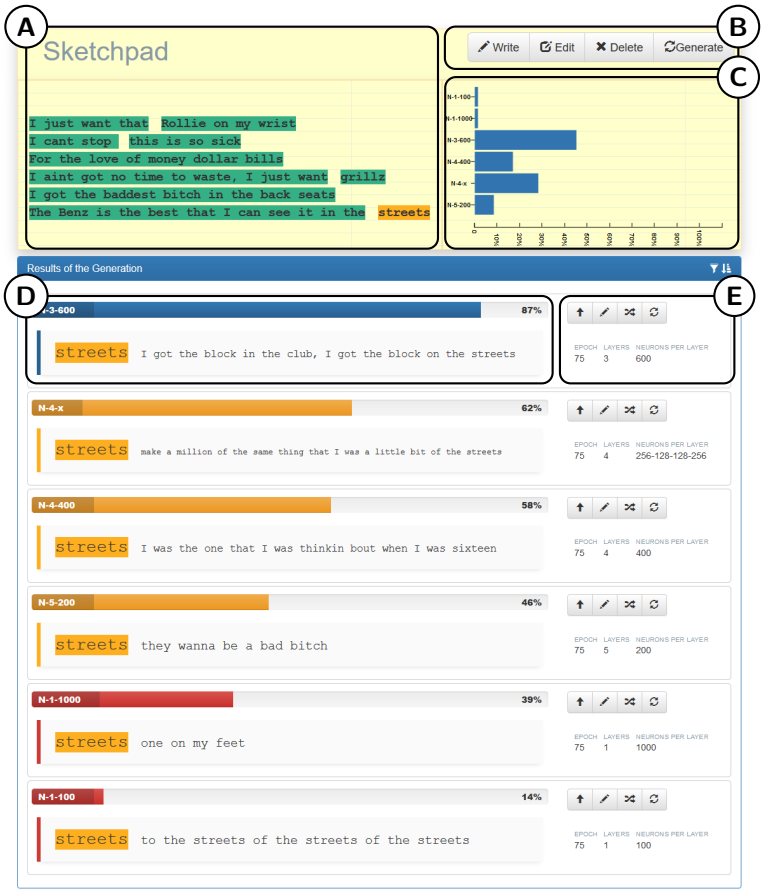

Fig. 2: **G-Rap Interface** (A) Sketchpad for the overall generated lyrics, (B) control panel for all RNNs, (C) selection statistics, (D) sketchpad for produced rap lyrics by a RNN with a ranking score, (E) control panel for a specific RNN.

model sequential data, without attention extension. LSTM layers enable storing information through a larger period of states and allow the network to use the previous states of a sequence to predict a possible future state.

G-rap was used to evaluate exemplary multiple RNN architectures which have one to five layers with a constant and also varying number of neurons per layers. Fig. 2 (E) shows, which configurations and training epochs were chosen. These exemplary architectures consist of two one-layer networks with 100 and 1000 neurons, a three-layer network with 600 neurons per layer, a four-layer network with 400 neurons per layer a five-layer network with 200 neurons per layer, and a network with 4 layers and 256, 128, 128, 256 neurons. They are trained on a Tesla P100 for 75 epochs.

To support the user in the selection process, we provide basic quality measurements as a score and a ranking for the generated line. In the application case, this is a combination of a rule-based style and grammar checker, which emits a score for the ranking process.

The G-Rap workflow begins with the generation of random lines for each RNN or with a custom user provided input for all RNNs or a particular RNN. This can be done with the overall sketchpad (Fig. 2A) using the control panel for all RNNs (Fig. 2B). Further, each RNN has also a smaller sketchpad (Fig. 2D) and control panel with an architecture description of the RNN (Fig. 2E). The score for each produced line is included in the sketchpad via a progress bar(Fig. 2D). After the generation and custom modification of lines, the user is able to copy the best result to the overall sketchpad. By iteratively generating and using different results the user is able to create novel lyrics. The selection statistic(Fig. 2C) presents the number of selected lines from each RNN. These statistics help to evaluate the global performance of each RNN, in other words: how often it was selected as best output by the user.

3.2 Examples using G-Rap

To demonstrate the practicality of G-Rap, we collected around 120.000 rap songs from 955 English speaking artists, who combine various styles, techniques, and vocabulary in different sub-genres like R&B and Trap. The following lyrics are examples we generated and concatenated with G-Rap from different RNNs.

For the love of money dollar bills We just getting started dont panic When I wake up in the morning I can see the sun come up	I can see the stars in the sky I can see the sun shine and she stayed in my eyes She said she want to see me shine

4 Future Work

We presented G-Rap, an interactive approach to iterative text synthesis with recurrent neural networks. Contributions include the capability to determine the best suitable RNN for certain tasks even if the output cannot be judged automatically. Further, by comparing the results of arbitrary RNNs with the same input, G-Rap enables users to progressively build the desired output in a productive workflow. Basic quality measures and network selection statistics provide context for the result selection.

While we have shown examples for the applicability of our approach using rap music lyrics, we envision the concept of G-Rap to be adapted to further application scenarios. Suitable use cases could be outputs where a) quality cannot be evaluated automatically and b), the generation process is repetitive. Examples comprise text syntheses such as poetry generation and object detection and localization in images.

We intend to build on this initial publication in several ways: First, while only conceptually mentioned so far in Fig. 1, implementing the feedback loop for

selected results back into the RNN training cycle via active learning could help to improve complex outputs, e.g. images. Second, since the number of network outputs a user can efficiently observe for one iteration is limited, we aim to assist the user by filtering the number of displayed network results. Finally, a better control over the amount and selection of training epochs could help to detect overfitting issues transparently in the workflow.

References

[1] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[2] M. Ghazvininejad, X. Shi, Y. Choi, and K. Knight. Generating topical poetry. In *EMNLP*, 2016.

[3] M. Ghazvininejad, X. Shi, J. Priyadarshi, and K. Knight. Hafez: an interactive poetry generation system. *Proceedings of ACL 2017, System Demonstrations*, 2017.

[4] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8), 1997.

[5] D. A. Keim, J. Kohlhammer, G. Ellis, and F. Mansmann. *Mastering The Information Age - Solving Problems with Visual Analytics*. Eurographics, 2010.

[6] C. Lamb, D. G. Brown, and C. L. Clarke. A taxonomy of generative poetry techniques. *Journal of Mathematics and the Arts*, 11(3), 2017.

[7] Q. Ma, L. Zhang, and B. Wang. New strategy for image and video quality assessment. 19, 2010.

[8] E. Malmi, P. Takala, H. Toivonen, T. Raiko, and A. Gionis. Dopelearning: A computational approach to rap lyrics generation. *arXiv preprint arXiv:1505.04771*, 2015.

[9] R. Manurung, G. Ritchie, and H. Thompson. Using genetic algorithms to create meaningful poetic text. *Journal of Experimental & Theoretical Artificial Intelligence*, 24(1), 2012.

[10] H. G. Oliveira, R. Hervás, A. Díaz, and P. Gervás. Multilingual extension and evaluation of a poetry generator. *Natural Language Engineering*, 2017.

[11] H. G. Oliveira, T. Mendes, and A. Boavida. Co-poetryme: a co-creative interface for the composition of poetry. In *Proceedings of the 10th International Conference on Natural Language Generation*, 2017.

[12] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2002.

[13] P. Potash, A. Romanov, and A. Rumshisky. Ghostwriter: Using an lstm for automatic rap lyric generation, 2015.

[14] Q. Wang, T. Luo, and D. Wang. Can machine generate traditional chinese poetry? a feigenbaum test. In *Advances in Brain Inspired Cognitive Systems: 8th International Conference, BICS 2016*. Springer, 2016.

[15] Z. Wang, W. He, H. Wu, H. Wu, W. Li, H. Wang, and E. Chen. Chinese poetry generation with planning based neural network. 2016.

[16] K. Watanabe, Y. Matsubayashi, K. Inui, T. Nakano, S. Fukayama, and M. Goto. Lyrisys: An interactive support system for writing lyrics based on topic transition. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces*. ACM, 2017.

[17] R. Yan. i, poet: Automatic poetry composition through recurrent neural networks with iterative polishing schema. 2016.

[18] R. Yan, H. Jiang, M. Lapata, S.-D. Lin, X. Lv, and X. Li. i, poet: Automatic chinese poetry composition through a generative summarization framework under constrained optimization. 2013.

[19] X. Zhang and M. Lapata. Chinese poetry generation with recurrent neural networks. In *EMNLP*, 2014.