Conversing verses - haiku generation using a LSTM-based auto-encoder matching model

Luka Ernestini
Univerza v Mariboru
Fakulteta za elektrotehniko, računalništvo in informatiko
Maribor, Slovenija
luka.ernestini@student.um.si

Niko Uremović
Univerza v Mariboru
Fakulteta za elektrotehniko, računalništvo in informatiko
Maribor, Slovenija
niko.uremovic@um.si

ABSTRACT
Lorem ipsum

Categories and Subject Descriptors
H.4 [**Information Systems Applications**]: Miscellaneous

General Terms
Theory

Keywords
text generation, neural networks, LSTM

1. INTRODUCTION
Deep learning is an emerging machine learning approach that can already be seen applied in many industries, including natural language processing. Example use cases include identifying people and objects in images and videos, understanding voice commands (smartphones, cars, smart houses) and providing better results for internet search queries. This technology brought machines the closest they have ever been to how we humans think and talk. Having learned the rules and grammar of our natural language, machines are now able to generate text for various applications that is in some cases indistinguishable from human written text [7].

In the field of natural language processing, many methods deep learning have been developed for text generation. Song, Huang and Ruan [9] achieved Abstractive Text Summarization (ATS) using and LSTM-CNN based ATS framework (ATSDL). You et al. [10] researched ways to generate natural language descriptions of images. They proposed a model of semantic attention which combines the established top-down and buttom-up approaches. CNN is used for image classification, followed by LSTM-based RNN for the caption generation. Bhagavatula et al. [2] investigated the viability of natural language-based abductive reasoning.

One of the obstacles on the field of text generation has been the objective evaluation of the generated text's semantics. While the spelling and grammar correctness are easy to evaluate, the meaning of the generated text can be challenging to evaluate with an automated metric [3]. Some tasks of text generation, such as image captioning or machine translation, where the (suggested) correct generation is known, can use metrics like BLEU [6] or METEOR [1], to evaluate the quality of the generated text. On the other hand, tasks where the correct generation is not specified (e.g., story [7] or poem [11] generation), the quality of the produced text is usually evaluated by human readers.

We draw inspiration for our research from the work of Luo et al. [4], who propose a novel Auto-Encoder matching model to learn utterance-level semantic dependency for generating everyday dialogue. Three neural networks are used: LSTM for encoding a reply into it's semantical representation, then a feedforward network for mapping the reply semantic into the answer semantic. Lastly a LSTM decoder is used for the sentence generation. Additionally, we found motivation for our work in the success on the field of poem generation by Potash, Romanov and Rumshisky [8], who demonstrated the generation of rap songs using LSTM, and Netzer et al. [5], who explored the usage of Word Association Norms (WAN) to generate Haiku poetry.

In the paper, we present a novel method for Haiku poetry generation, combining the approaches of generating the first verse using GRU network, with the Auto-Encoder matching model [4] for generating the second and third verse of the Haiku poem as a form of conversation between the verses. RNN, such as GRU or LSTM has proven to be capable of mimicking rhythm, common phrases and even poet's writing style [8], which we attempt to use for generating the text in a genuine Haiku style, which we describe in the continuation of the work. By using the Auto-Encoder matching model to generate the second and third verse as an answer to the previous verse, we experiment by adding a sense of conversational bond between the verses.

2. BACKGROUND
Haiku is a form fo poetry of poetry originated in Japan and later adopted in Western languages. Traditional Haiku consists of three lines with five, seven and five syllables, however this form has been loosened in non-Japanese versions

of Haiku. We adopt this loosened form of Haiku, with only syntactic constraint being 3 short verses.

For the training corpus, we selected approximately 17 thousand poems from a larger Haiku dataset, containing over 118 thousand English Haikus. Selected Haikus comply with our loosened syntactic constraint, wereas other poems in the dataset were of more experimental nature (empty or very long verses, unconventional use of punctuation, grammatically flawed).

3. METHODOLOGY

The proposed methodology of generating Haiku poetry consists of the following steps: generating the first verse of the poem using a GRU-based model, and generating the second and third verse using the modified Auto-Encoder matching model. The poem generating framework thus consists of 5 neural networks of 2 types. First type is the auto-encoder GRU-based network with 2 GRU cells. First GRU cell processes the verse and encodes it in a hidden state, which is then decoded by a second GRU cell that generates the output from the given state. 3 such networks are used in the framework, one for each verse. The second type of the neural network is a mapping network, that transfers the information from one verse to another. The mapping network consists of 2 fully-connected layers. The architecture of the auto-encoder model type is presented in Figure 1, and the architecture of the mapping module in Figure 2.

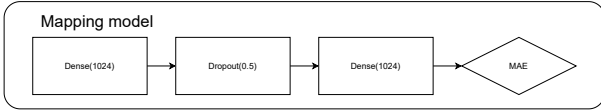**Figure 1: The architecture of the auto-encoder model type.**

Figure 2: The architecture of the mapping module.

The procedure of generating the Haiku with trained models is more precisely presented in the Figure 3.
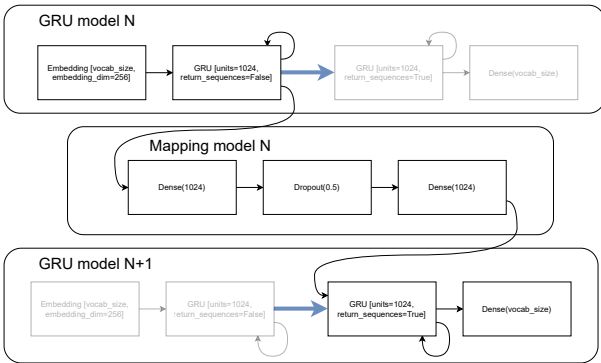

Figure 3: The procedure of generating the Haiku with trained models.

While training the GRU models is fairly straightforward (the auto-encoder model is supposed to generate the same output as the input it was given), the training of a mapping module is somewhat more complicated. Authors [4] of auto-encoder matching model for dialogue generation, train the mapping module by passing the encoded hidden state of an auto-encoder into a mapping module, and then decoding the mapped hidden state with auto-encoder and comparing it to the expected output (reply in a conversation). While we could in theory do the same (encode verse 1 with auto-encoder for 1st verse, map the hidden with mapping module 1 and then decode the mapped hidden state with auto-encoder for the 2nd verse, similar for the following 3rd verse), this is a very computationally difficult task. While we could pre-process the encodings of each verse, we would still have to do all the decoding operations with each training step, leading to very long training times. To mitigate this, we propose a simpler approach presented in Figure 4. Instead of decoding the mapped hidden state, the mapped hidden state itself is compared to an estimation of the expected hidden state. This approach offers significantly faster training times, however the exactness of the approach is not optimal, which we further discuss in the following sections.
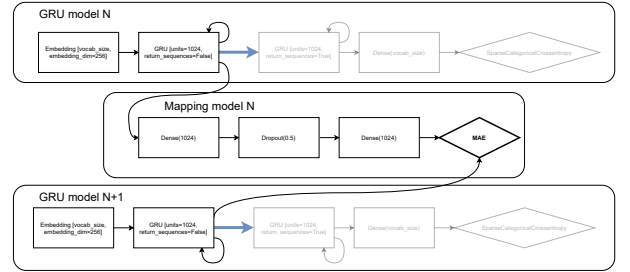

Figure 4: The procedure of training the mapping module.

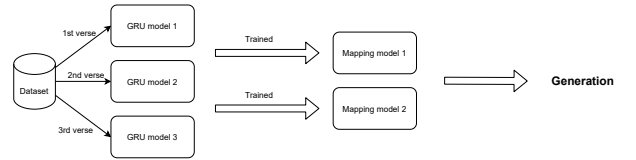The whole framework is presented in the Figure 5.

Figure 5: The Haiku poem generation framework.

4. RESULTS
5. CONCLUSIONS

We present a novel method for poem generation, applied to the field of Haiku poems. The method utilizes auto-encoder models based on GRU for learning poem characteristics on verse-basis and a mapping module for propagation of information between the verses. A sample of generated poems were together with a sample of human-written poems presented to a number of respondents, that evaluated both generated and human-written poems. While an average generated poem was evaluated 27% worse than an average human-written poem, there were 3 outstanding poems generated that were evaluated as good as one of the best human-written poems in the sample presented.

The results could be improved by ommiting the training cost optimization explained in section Methodology. Two factors

that make this approach inexact, are the following: simply comparing the mapped multi-dimensional hidden state to so-called "estimated expected hidden state" with metric MAE is far from optimal, since the importance/sensitivity of each element in a hidden state vector is not the same. Secondly, the "estimated expected hidden state" is as the phrasing suggests, just an estimation. There can be a number of hidden states that would decode into a target verse, and finding such sub-space of hidden states (if it even exists - if not, we should seek for closest matching verse), is computationally even more difficult than decoding the state in each step of training.

6. REFERENCES

[1] S. Banerjee and A. Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, 2005.

[2] C. Bhagavatula, R. L. Bras, C. Malaviya, K. Sakaguchi, A. Holtzman, H. Rashkin, D. Downey, S. W. tau Yih, and Y. Choi. Abductive commonsense reasoning, 2020.

[3] A. Celikyilmaz, E. Clark, and J. Gao. Evaluation of text generation: A survey, 2020.

[4] L. Luo, J. Xu, J. Lin, Q. Zeng, and X. Sun. An auto-encoder matching model for learning utterance-level semantic dependency in dialogue generation, 2018.

[5] Y. Netzer, D. Gabay, Y. Goldberg, and M. Elhadad. Gaiku: Generating haiku with word associations norms. In *Proceedings of the Workshop on Computational Approaches to Linguistic Creativity*, pages 32–39, 2009.

[6] 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 of the Association for Computational Linguistics*, pages 311–318, 2002.

[7] D. Pawade, A. Sakhapara, M. Jain, N. Jain, and K. Gada. Story scrambler-automatic text generation using word level rnn-lstm. *International Journal of Information Technology and Computer Science (IJITCS)*, 10(6):44–53, 2018.

[8] 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*, pages 1919–1924, 2015.

[9] S. Song, H. Huang, and T. Ruan. Abstractive text summarization using lstm-cnn based deep learning. *Multimedia Tools and Applications*, 78(1):857–875, 2019.

[10] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo. Image captioning with semantic attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[11] X. Zhang and M. Lapata. Chinese poetry generation with recurrent neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 670–680, 2014.