Generating Haiku using Generative Adversarial Network

Lakshay Gopalka, Ziwei Lin, Ran Jing, Hanwen Hong
School of Engineering
Worcester Polytechnic Institute
Worcester, MA — 01609-2280

***Abstract*—Haikus are short Japanese style poem which includes a set of phrases connected to form an artistic verse. The elements in Haiku plays an essential part and therefore, to preserve the semantics, we propose a machine learning method to generate them. We use neural networks (NN) and Generative Adversarial Networks (GAN) to generate Haiku and test the quality of the produced Haikus using performance measure. GAN uses a discriminative model to guide the training model to output real-valued data. We overcome the difficulties of conventional GAN by using RelGAN, LeakGAN and SeqGAN for text generation applications as bypasses the generator discriminator by directly performing the gradient policy update. Extensive experiments using various network topologies were conducted on the dataset to demonstrate the effectiveness of GAN models over the NN models by evaluation measures like perplexity and BLEU.**

***Index Terms*—Haiku, Generative Adversarial Network, Neural network, Machine learning, Text generation.**

I. Introduction

Haiku is a type of short poem originated in Japanese culture in the early 17th century. A typical haiku is written in Japanese language but with the recent modernization, the Haikus are obtaining popularity in the English language. A famous example of Haiku in the English language is-

An old silent pond...
A frog jumps into the pond...,
splash! Silence again.

The traditional Haiku is composed using two basic rules namely, three phrases having 5, 7 and 5 syllables in each phrase. Thus, a total of 17 syllables and a Haiku should consist of a seasonal word like in the example given, frog represents springs in the Haiku.

The process of narrative generation for Haiku involves accompanying a virtually small story and interpretation of the results. The project aims to generate Haiku using neural and adversarial networks so the generated text output can gradually transform from a series of phrases to meaningful Haiku. The text generation and assistance from Artificial Intelligence networks is a challenging part and for our application, the quality assessment needs to be done not only based on grammatical correctness but also using lyrics and flow in the Haiku verse [4]. It is important as a lot of applications like text translation, data sampling, chatbots etc, are dependent on text modelling and generation. The deep neural networks can produce results which cannot be put just by putting characters but include innovation and a new style. These systems also assist human writers to make the writing process effective and efficient. As a result, these systems are already been employed in Industrial sectors for daily purpose applications. The propose of our project is to explore the potential of learning-based algorithm and compare the difference of the output results.

To achieve the goal, we are using a deep neural network to generate Haiku in the English language. The project aims to resolve and generate a new method for producing text or poems using a network topology, which will not only produce a three-phase Haiku but also generate artistic features. The text generation involves correct semantic, structure, accuracy, rhyme, and other several features. The project tries to look over the following considerations to output a well-formatted Haiku using text modelling and generation using machine learning techniques. Another motivation behind this project to make people familiar with traditional culture and integrating Haiku along with modern AI process could spark an interest among the newer generations to know more about the learning process.

The process of narrative generation involves rigorous training of the model and network, feed the training data appropriately and have an assessment protocol which evenly evaluates the data. The challenge involves ensuring a correct network, so the system does not over-fit or under-fit in any case, to retain the characteristics of the English language and haiku. The short poems should make sense to the readers and should blend with accompanying two phrases. The two main approaches to solve the text generation theme and issue is to model a neural and adversarial network. Our approach uses Recurrent neural network (RNN) along with Generative adversarial network (GAN) to optimize the algorithm. Furthermore, we employ the variations of the network to compare the results, like the seqGAN or sequential GAN for training which we will discuss in the next sections.

Employing the machine learning process marks the improvement of modern technology and the development of computing fields from medieval times. Using such a methodology for generating English Haiku signify the relationship of humans with their culture and to produce an impact for the society. This aims to bring more diversity and closeness among people by popularizing such art form, like Haiku. It also reflects the growth of the AI field as we apply deep learning and machine learning methods like GAN which fits the structure of a language appropriately and the evaluation method like f-measure, BLEU complements our hypothesis.

II. Background

Generating meaningful text is a critical topic in machine learning studies and many efforts have been performed to contribute to this domain. In 2011, a mutation of Recurrent-Neural-Network with multiplicative connections has been applied to generate text[12]. The paper presented an algorithm which made a significant step on character level learning and text generating. The Generative Adversarial Nets were invented by Ian Goodfellow in 2014, which is a novel powerful tool for deep generative models and improved the learning process significantly[2]. A mutation of this algorithm with leaked information is used to create a meaningful short paragraph that contains more than 20 words[3]. These topologies use a discriminate model to guide the training of generative model which obtained reasonable success in producing output in resemblance to training data. These models can assess the complete network sequence and the two networks compete with each other to display real-valued output. Using such methods, some researchers have successfully implemented these networks in the model generation and producing synesthetic data for their applications. Recently, a group of researchers also managed to generate a formatted poem by setting up a learning model which allowed the agent to predict the next word with the given input. Hence, we are using such state-of-art machine learning methods to extract out the Haiku for our study.

The concept of poem generation is a little different[6] in the current generation as the goal is to mimic the content but in an expressional form. The traditional Japanese haiku and the text generation work has been researched by Xianchao Wu and his colleague with the implementation of a SeqGAN in 2017[13]. This was very significant and promising, but it does not perfectly fit in this situation due to different expectations. Applying these GAN networks have two main issues. The foremost being, GAN is designed in a way to generate real value continuous data but has difficulties in producing discrete based outputs and directing a sequence of results like texts. Secondly, it is only able to give the score or loss for an entire function sequence and it is uncertain to measure its performance for a partially generated sequence[14]. Nevertheless, GANs have outperformed existing networks as they offer several advantages over other methods of supervised or unsupervised learning. They do not require explicit labelling of training data as they are an unsupervised learning method. They can generate new results which are like existing real-valued data and have the ability to learn the distribution of data. Finally, the discriminator network is a classifier which can be used to classify objects as the GAN network has two models: a discriminator and a generator.

The GAN models are diverse in nature and variation of these models like SeqGAN, RelGAN exist to produce a variety of outputs for different kinds of applications. Relational GAN (RelGAN) was introduced in 2017 to text generation domain. This method represents the state-of-the-art of deep neuron nets as it employs the relation memory system to replace the Long short-term memory (LSTM) in normal memory-GAN algorithm[9]. This GAN mutation also does not heavily depend on character-wise input, which is ideal for the character- syllables application in Haiku. For the purpose of discrete output, we use the generative model as stochastic property policy which avoids the difficulty for discrete data in conventional GAN. Also, the discriminator is used to evaluate the sequence and give feedback evaluation to guide the learning of the generative model. Thus, we chose this learning method as a possible solution for the project.

Paper Organization: The Introduction section summarizes about the Haiku and provides an overview of the problem. The section–II covers the related work of other researchers and introduces GAN topology. The rest of this paper is organized as follows. Section III provides an overview for data collection and data set process, explanation of the algorithms used along with experimental design for each network. Section IV explains the evaluation method for calculating the accuracy for learning models and describes the results produced. Section V and VI discusses the conclusion and findings of the project and talks about future scope of the problem.

III. Method

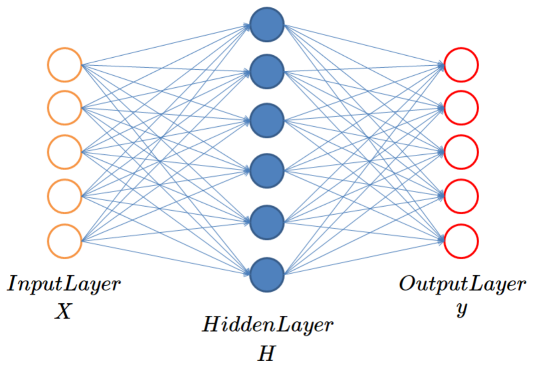
A. Data sets

The data source is an extremely essential parameter and as the dataset of Haiku is not explicitly available on data source websites like Kaggle or Google data search option. Therefore, the dataset is gathered using crawl algorithm which extracted the Haiku from different websites containing Haiku archive in the English language. The implementation is completed using Python libraries Beautifulsoup and Newspaper3k.

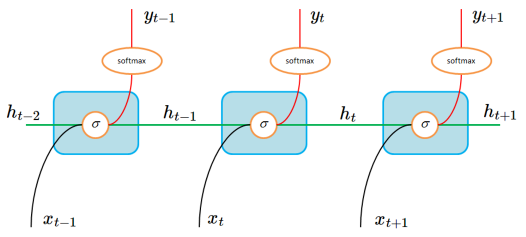
The collected data is refined to maximize the performance of the algorithms. All the additional or special characters are removed from the dataset. We also removed any foreign words as it increases the cost of syllables and hinders with the learning process. The alphabets in the Japanese language contain one syllable and each kanji has one to two syllables. As a result, the Japanese Haiku, when following a 5-7-5 syllables rule, will always have a form with a combination of 3-5 kanji per lines. However, those correlations do not exist in English. For example, both 'amazing' and 'friends' have six characters, but the first one has three syllables while the 'friends' only have one. Hence, because of this trait, English Haiku writers believe the syllables should not be limited to 5-7-5 but in form short-long-short and we observed this in the collected dataset. To evaluate this further, we wrote a script to count the syllables for each Haiku in our data set and found less than 1 percent of haiku has 17 syllables. Hence, we use the short-long-short form for our purpose.

Also, Haiku generation is different from normal long text generation as the poem does not necessarily need a period at the end of lines. But both RNN and GAN use a character-wise input vector which means a complete Haiku will be one instance that will be sent as one training data. Therefore, in our case, we add a special symbol after each line to represent a three lines design. This will also enable the AI model to explicitly obtain a good range of vocabulary for composing and creating new Haiku for the project's purpose.

The dataset is expected to contain at least 6000 well-written Haiku with a verse English style and pose. In the following method and experiment design part, more details will be presented as to how we use the data in the learning algorithm.



(a) The framework of minimal RNN: RNNs can take one or more input vectors and produce one or more output vectors and also have a "hidden" state vector based on input(s)/output(s)



(b) Initializing the layer and hidden layers which will loop through inputs, pass the word and hidden state into the RNN and returns the output modified hidden state.

Fig. 1: RNN framework description

B. Algorithm

1) RNN and Minimal-RNN: RNN (Recurrent Neural Network) is commonly used in the area of natural language processing. The framework of RNN is shown in Fig.1(a). From the figure, we can define the weight from input layer to hidden layer as W_{xh},the weight from one hidden layer to another as W_{hh},and the weight between the final hidden layer to the output as W_{hy}. The state of hidden layer at time step t is h_t. The basic idea of RNN is to stack one or more hidden layers of previous time steps, and each hidden layer depends on the corresponding input at that time step and the previous time step[8], which is shown in Fig.1(b). The hidden layer is described in:

$$h_t = f\left(W_{xh}x_t + W_{hh}h_{t-1}\right) \tag{1}$$

And the output is computed using only the associating hidden layer:

$$y_t = softmax\left(W_{hy}h_t\right) \tag{2}$$

We are implementing Char-RNN, which uses characters (letters) as the input for each time step and Minimal-char-RNN is the simplest version of char-RNN, which has only two hidden layers in the network[12]. The advantage of RNN is the strong capacity to capture contextual information. However, RNN is a biased model (i.e. later words are more dominant than earlier words). Thus, effectiveness reduces when it deals with whole documents or long sentences.

2) Long Short Term Memory (LSTM): Long Short Term Memory (LSTM) is modified version based on RNN. LSTM model proposes to use a LSTM layer to represent each sentence and then passing another RNN variant over these layers[11]. To some extent, the LSTM model can solve the gradient vanishing problem of RNN. It adds a forget gate to keep some information from the previous cell state. The framework of LSTM is shown in Fig.2.

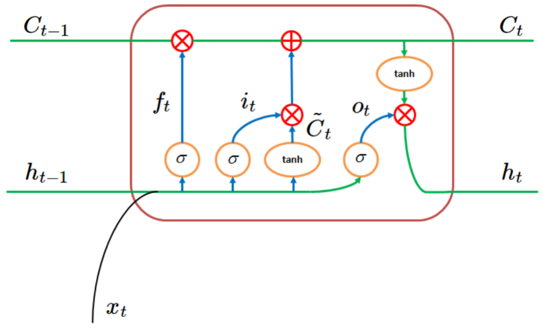


Fig. 2: The Framework of LSTM

LSTM perform sequence-to-one mapping where tanh represents prediction based on hidden layer, σ is the element-wise sigmoid function, i_t , f_t , o_t and C_t are respectively the input, forget, output gate and the cell activation vectors.

The mathematical model for RNN is described below: x_t is the input at current time-step t, h_t is the hidden layer value at current time-step t. i_t, f_t,and o_t are the input gate activation, forget gate activation and output gate activation respectively. W_i, W_f and W_o are weight values for the input, forget and output layer respectively. σ represents the logistic sigmoid function. From what RNN could accomplish, we could simply get:

$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right) \tag{3}$$

Comparing to RNN, the equation above is exactly the same with RNN to compute the hidden state at time step t. But it's not the actual hidden state in terms of LSTM, so it is named as o_t. So from here, we will see how LSTM is improved from RNN.
First, LSTM is given the ability to "forget", which means it can decide whether to forget the previous hidden state and is implemented by adding Forget Gate Layer:

$$f_t = \sigma\left(W_f \cdot \left[h_{t-1}, x_t\right] + b_f\right) \tag{4}$$

In contrast to the forget gate layer and to tell the model to update current state using the previous state, we need to add Input Gate Layer accordingly:

$$i_t = \sigma\left(W_i \cdot \left[h_{t-1}, x_t\right] + b_i\right) \tag{5}$$

Then we will compute the temporal cell state for the current time step. It looks just like the output of RNN above, except that $tanh$ activation function is used. After we have computed the current cell state(c_t), we will use it to compute the current hidden state(h_t) and final output(y_t).

$$h_t = o_t * \tanh\left(C_t\right) \tag{6}$$

$$y_t = softmax\left(W_{hy} h_t\right) \tag{7}$$

3) SeqGAN: Sequence Generative Adversarial Networks (SeqGAN) is a variation of GAN with the difference in generator network as it as an RL agent[14]. The policy is trained using policy gradient as the generator is regarded as stochastic parametrized i.e. the actions are drawn from distributions. The model generates a sequence to maximizes the expected reward which is evaluated by the discriminator. Thus, the algorithm runs recursively until the SeqGAN produces output validated by the discriminator.

4) LeakGAN: Leak Generative Adversarial Networks(GAN with Leaked Information, LeakGAN) was developed in 2017 [3] to address the problem for long text generation with GAN. The novel part of this algorithm is that the generator is extracting useful information into all generation steps through an additional manager module, which takes the extracted features of current generated words and outputs a latent vector to guide the worker module for next-word generation. The framework of LeakGAN is shown in figure 3.

5) RelGAN: Relational Generative Adversarial Networks or (RelGAN) is one of the latest GAN network which is proposed in 2019.[9] Thethought behind this GAN network is that it applied the concept of Gumbel-Softmax relaxation.[5][7]

[1]https://github.com/CR-Gjx/LeakGAN/blob/master/figures/leakgan.png
[2]https://github.com/williamSYSU/TextGAN-PyTorch/blob/master/assets /model_relgan.png

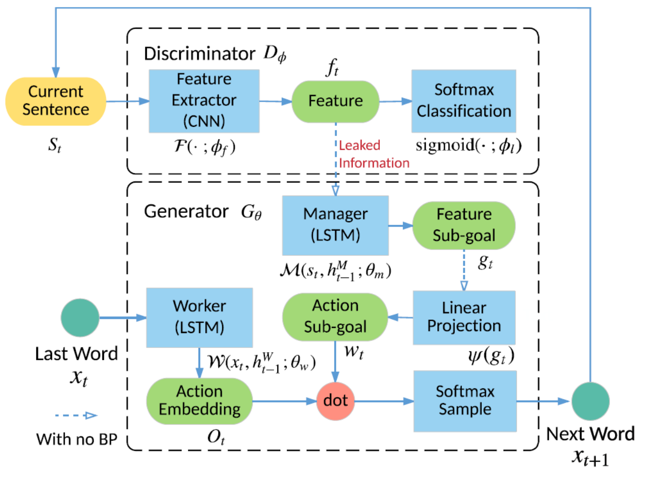


Fig. 3: The framework of LeakGAN[1]
While the generator is responsible to generate the next word, the discriminator adversarially judges the generated sentence once it is complete. The chief novelty lies in that, unlike conventional adversarial training, during the process, the discriminator reveals its internal state ,feature f$_t$ in order to guide the generator more informatively and frequently.

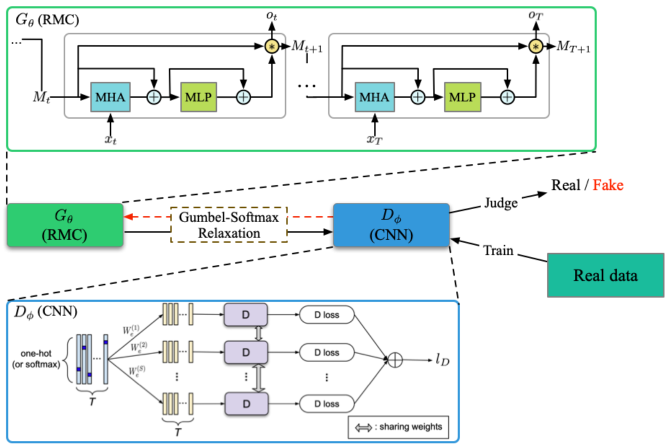


Fig. 4: The framework of RelGAN.[2]
The self-attention mechanism for updating the memory by incorporating new observation and softmax function is performed on each row.The proposed discriminator framework which has input as either the real sentence or the generated sentence.

As shown in figure 4, the Gumbel-Softmax relaxation is designed for passing the generator loss from discriminator to generator. Gumbel-Softmax relaxation includes Gumbel-Max trick and discreteness relaxing. Gumbel-Max trick is the one-hot embedding of standard Gumbel distribution. Meanwhile, the relaxing will produce a replacement of one-hot embedding that it includes a new parameter: tunable inverse temperature. The inverse temperature is a key parameter in Gumbel-Softmax relaxation. With large inverse temperate, the variance of gradients will be very large, this may cause very poor sample quality. However, with small inverse temperature, we will have a larger(initial) approximation gap between the generated one and the real one, which will lead the generator to a more ”Conservative” generating strategy. In our experiment, we set the initial temperate to 100 in order to get a decent result.

C. Assessment protocols

We are using BLEU(bilingual evaluation understudy) and perplexity to measure the performance of our language model and other classic approaches.

1) BLEU: BLEU is an algorithm to evaluate the quality of the text [10]. It is originally used in the language translation algorithm but in our method, we are using it to evaluate the performance of our generative model. BLEU calculates scores that are averaged over the whole corpus to reach an estimate of the translation/generation’s overall quality. It is a modified form of precision to compare a generator’s output ($C = c_1, c_2, ..c_i$) against multiple references.($S = s_1, s_2, ..s_i$),and $h(c_i)$ represents the frequency for the word to appear in the output series. The way to calculate is as follows:

$$CP_n(C,S) = \frac{\sum_i \Sigma_k \min\left(h_k\left(c_i\right), \max_{j\in m} h_k\left(s_{ij}\right)\right)}{\sum_i \Sigma_k h_k\left(c_i\right)} \tag{8}$$

In summary, BLEU counts the frequency of the words that appear both appear in the generator and the references and multiply them together to find probability of the sentence and asses its correctmness. BLEU score is calculated as:

$$BLEU_N(C,S) = b(C,S)\exp\left(\sum_{n=1}^{N} \omega_n \log CPn(C,S)\right) \tag{9}$$

2) Perplexity: After learning the distribution of the language model from the training Corpus, we can calculate the perplexity to measure its quality [1]. If the test corpus is ($\mathcal{E}_{test}$), and each statement in the test corpus is a sequence of words $E = e1, e2, ..e_m$, then:
Per-word log-likelihood calculation is described as:

$$WLL\left(\mathcal{E}_{test}\right) = \frac{1}{\sum_{E\in\mathcal{E}_{test}} |E|} \sum_{E\in\mathcal{E}_{test}} \log_e P(E) \tag{10}$$

Per-word cross entropy calculation is described as:

$$H\left(\mathcal{E}_{test}\right) = -\frac{1}{\sum_{E\in\mathcal{E}_{test}} |E|} \sum_{E\in\mathcal{E}_{test}} \log_2 P(E) \tag{11}$$

Perplexity calculation is described as:

$$ppl\left(\mathcal{E}_{test}\right) = 2^{H\left(\mathcal{E}_{lest}\right)} = e^{-WLL\left(\mathcal{E}_{last}\right)} \tag{12}$$

In which, $P(E)$ is the language model.

$$P(E) = P\left(e_1 \cdots e_m\right) = \prod_{i=1}^{m} P\left(e_i | e_1, \cdots, e_{i-1}\right) \tag{13}$$

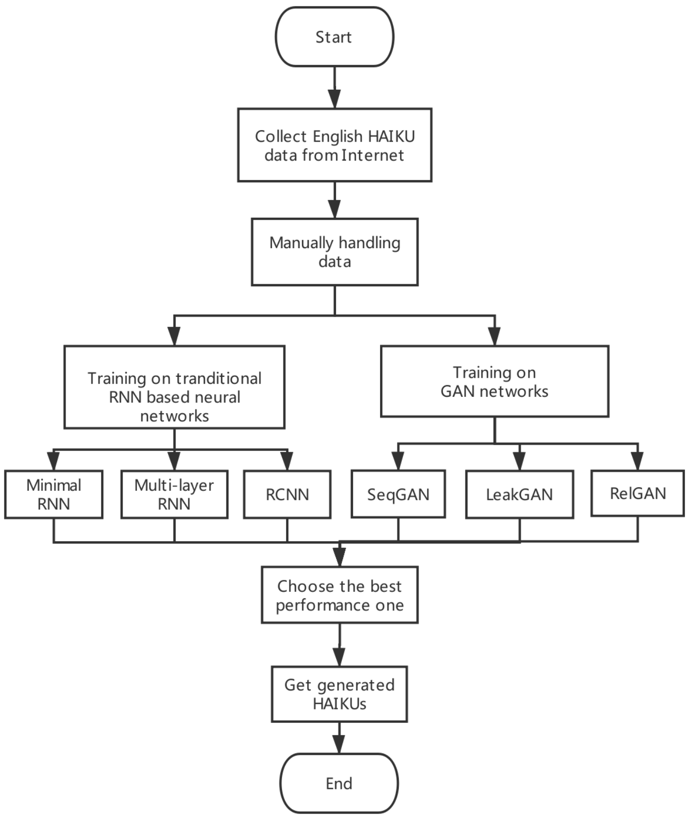


Fig. 5: The framework of our experiment design

D. Experiment Design

As shown in figure 5, we applied six different neural networks in our experiment framework in order to generate Haikus. First, we collected thousands of Haikus from Internet and filter them to get proper English Haikus. Then, we use long-short-long term to check Haikus and convert some Haikus into one line so that they can be trained properly. Finally, we performed the text generation using different methodology and chose models with different parameters to get best performance for generating Haikus.

IV. Result

We trained all six algorithms following our experiment design with the data set. During the experiments, we noticed minimalRNN, SeqGAN and RCNN algorithms are easily reaching an over-fitting situation during the training section. As described above, we used two scores, BLEU-3 and Perplexity for evaluating and comparing those algorithms. For three GAN networks, it is hard to get the probabilities for every word, which makes it is hard to calculate perplexity results for GAN networks.Hence, we were unable to evaluate GAN using perplexity scores and there are only perplexity results for three RNN based algorithms. The BLUE-3 result for six algorithms and perplexity results for three GAN networks are described in accompanying tables.

TABLE I: BLEU-3 Algorithms Results:
BLEU-3 indicates we choose 3-gram based BLEU metric as our criteria. The Higher score means the generated Haikus are more similar to the original Haikus. In this case, LeakGAN outperforms all the other networks.

Algorithm	BLEU-3
MinimalRNN	0.0
MultiLayerRNN(LSTM)	0.0121
RCNN	0.0479
SeqGAN	0.404
LeakGAN	**0.533**
RelGAN	0.357

TABLE II: Perplexity Results:
Perplexity scores are measure of the network ability to generate similar data as training data and lower perplexity score means a superior model.In this case,RCNN outperforms other neural networks.

Algorithm	Perplexity
MinimalRNN	814.03
MultiLayerRNN(LSTM)	309.20
RCNN	**127.30**

After achieving the formula based evaluation scores for various tested algorithms, we start to work on generating sample Haikus. We employed Haikus from the three mentioned neural networks and GAN models. The following are some of the results of each network:

MinimalRNN:

in moon
a sumpald ants lanbseidst
in in wowd wy the fucny

moose
liveey wardew in latht
sallom tha cray on til

MultiLayerRNN:

winter night
he hears the scent
of cane doestricts

overcast
a new mail
on my coffee cup

RCNN:

subway skies
the long of shadow
runners

crescent travels ...
in the lamp storm
black all of the storm

SeqGAN:

autumn snow –
she takes me on the
well fed

half moon –
dreams starts the last
thoughts of day

LeakGAN:

globe gazing
fogging the pale moon
in the snow

a slow sun
the faint rose
in the red

RelGAN:

memorial day
the flowers have long faded
on the last rose

crescent moon
just one more
for the wind

The results for various neural and adversarial networks exhibits the potential of the AI field. The generated output rhymes and compares well with traditional Haikus for most part. Through the observation, we found that GAN network are better suited and produced a superior results over the neural network.

V. Conclusion

We successfully developed and implemented various machine learning methods to produce haiku. We worked on text generation domain and tried to broadcast Haiku to the community. The models are able to produce Haiku with decent accuracy. The RNN networks like RCNN outperforms among the NN learning methods discussed, which is found using perplexity while LeakGAN outperforms other topologies while evaluating using BLEU measures. The results using our network models are not perfect as some of the Haiku have grammatical errors or spelling mistakes but using GAN models have significantly enhanced the quality of the Haiku.

The resultant output from the GAN network is logically sound and free of any major errors. The proposed methodology has advantages over the neural network as it is better suited for text generation and considers various parameters required for semantic structure in case of text modelling applications. The GAN model was able to perform well however, a few areas on which our method could improve on includes the elimination of special characters to constitute the 3-line structure of Haiku. The model could have been better suited if it adapted to count the syllables in the Haiku and decide if it is acceptable using the discriminator network.

Overall, producing an artistic and poetic poem which relates to tradition and cultures using a modern computation algorithm is an indication of how far we have developed as a society. With the advent of learning methods, the scope and growth of such research areas will continue to rise to enhance the decision and quality of data on a larger scale.

VI. Future Work

As future work, we wish to conduct experiments with a larger number of Haiku dataset. We also found that the syllables were hard to handle in the English language. Though we obtained significant success in SeqGAN and LeakGAN models but we wish to train our model by using a pre-train system. It will provide a prediction based on the correlation between syllables and English text and implementing alongside the hidden layers with a syllable function, may result in 5-7-5 syllables Haiku form. Another direction to improve the output is to push a keyword system which will aid the algorithms to generate more meaningful poems on the given input topic. We can also use better assessment protocols methods and feedback of human writers and experts to improve the quality of output haiku.

References

[1] Stanley F Chen, Douglas Beeferman, and Ronald Rosenfeld. Evaluation metrics for language models. In *DARPA Broadcast News Transcription and Understanding Workshop*, pages 275–280. Citeseer, 1998.

[2] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[3] Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. Long text generation via adversarial training with leaked information. In *AAAI*, pages 5141–5148, 2018.

[4] Takuya Ito, Jumpei Ono, and Takashi Ogata. Haiku generation using gap techniques. In *Proceedings of the 2018 International Conference on Artificial Intelligence and Virtual Reality*, pages 93–96. ACM, 2018.

[5] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[6] S. Kikuchi, K. Kato, J. Saito, S. Okura, K. Murase, T. Yamamoto, and A. Nakagawa. Quality estimation for japanese haiku poems using neural network. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–8, Dec 2016.

[7] Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[8] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*, 2010.

[9] Weili Nie, Nina Narodytska, and Ankit Patel. RelGAN: Relational generative adversarial networks for text generation. In *International Conference on Learning Representations*, 2019.

[10] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[11] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. Lstm neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association*, 2012.

[12] Ilya Sutskever, James Martens, and Geoffrey Hinton. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 1017–1024, USA, 2011. Omnipress.

[13] Kazushige Ito Zhan Chen Xianchao Wu, Momo Klyen. Haiku generation using deep neural networks. *The Association forNatural Language Processing*, 2017.

[14] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.