Automated Haiku Generation based on Word Vector Models

Alham Fikri Aji

Master of Science

School of Informatics
University of Edinburgh

2015

Abstract

Poetry generation is a major challenge in artificial intelligence and computational creativity. Poetry generation requires a lot of linguistic resources. This requirement might cause difficulty in poetry generation in cases where such resources are not available. We purpose an automated Haiku generation based on Word Vector model. Word Vector model can be trained through normal unlabelled text corpus and can be used to capture word semantic and linguistic similarities. Our purposed system also uses Haiku corpus and Part-of-Speech tagger to capture unusual grammar patterns that often appear in poetry. Finally, this system uses bigram language model to capture the meaningfulness of poetry. We tested our system by using a survey. In this survey, we collect several random and selected generated Haiku and mix them with human-made Haiku. Respondents have to rate the Haiku meaningfulness, beautifulness, and topic relatedness. Additionally, respondents have to predict the writer of the Haiku. The result says that selected generated Haiku have similar meaningfulness and beautifulness score compared to real human-made Haiku. However, our randomly generated Haiku still have least score in both measurements. Both selected and randomly generated Haiku have a very good score in topic relatedness.

Acknowledgements

All praise and gratitude to God, the Almighty, for His mercy to give me everything to accomplish this work.

I would like to express my appreciation towards people who helped me through completing this work. Without their support and assistance this work would not have been possible.

- To my supervisor Dr. Daniel Winterstein for the insight, motivation, and supervision.
- To my parents in Indonesia. Even though they are not here and cannot support me directly, I believe their moral support helped me a lot through this study. Also, I would like to thank my wife for her continuous care and support.
- To DIKTI, which financially supports me with the scholarship. Without this scholarship, I would not have been able to pursue my master degree in the University of Edinburgh.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Alham Fikri Aji*)

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Poetry is one of the natural human methods to communicate by encapsulating the message in beautiful and artistic manner. William Wordsworth said that poetry as ”the spontaneous overflow of powerful feelings: it takes its origin from emotion recollected in tranquillity” [1]. More non-philosophic and restricted description of poetry is defined by Manurung as a text that meaningful, grammatically correct, and beautiful [2].

Automatic poetry generation is a challenge in artificial intelligence and computational creativity. It is considerably interesting, as it works on human creativity, emotional and intelligence domains [3]. The difficulties of this task come from various factors. One major issue lies in the poem evaluation. Evaluating poems and creative arts tend to be subjective, thus automated poetry generation faces difficulty obtaining objective result [4].

Writing poetry requires a lot of rewording and paraphrasing. Additionally, poetry are often flexible in grammar, lexical, and word-order. Therefore, it increases the space-search of poetry generation significantly[5]. Furthermore, it causes the definition of grammatically correct in poetry vague.

From those issues above, we conclude that a poetry generation system has massive resources requirements. Creating a poetry that satisfies diverse constraints such as phonetic, syntax, grammar or semantics requires wide arrays of resources [6]. For example, the system might need to know the text structure through a context-free grammar, understand the semantics representations or understand the meaning through WordNet[7]. However, such resources are not always available, especially in some unexplored languages.

To tackle vast amount of required resources to produce a poetry, we purpose Word Vector based poetry generator. Word Vector model can be trained by a standard unlabelled corpus and can be used to capture semantic relations of words. By narrowing down the required resources, we hope that our system can be adapted in Haiku and poetry generation in different language.

1.2 Objective

The goal of this work is to explore further about poetry generation. This work tries to answer the challenge in computational creativity on automatic poetry generator as one of the human artefacts. This work focuses on Haiku poetry.

The detailed objectives of this project is expressed as questions below

1. How to design Haiku generation system that utilises Word Vector model?
2. What are the required resources in building such system?
3. Can we generate Haiku with predefined specific keywords?
4. How good is the generated Haiku?
5. Can users differ between the generated Haiku with real human-made Haiku?

1.3 Methodology

This section presents the methodology used in this work. In general, the working flow of this project consists of literature review, experiment and algorithm design, data collection, implementation, and evaluation.

Explanations of each step are described as follows:

1. Literature Review

   In this part, various literature was studied. We explore various relevant information required such as the theories of Haiku, natural language processing, and previous related works.

2. Experiment Design

   In this part, the approach and the method that will be used to generate Haiku are designed. The experiment flow of this work is also designed in phase.

3. Data Collection

   In this step, we collect required resources for this project, such as Haiku corpus, word dictionary, word vector, or other resources.

4. Implementation

   The system is implemented in this phase, with a predefined design, resources and constraint obtained from the previous steps.

5. Evaluation

   The result of our system is tested and evaluated in this phase by using an online survey to measure our Haiku with certain criteria. Furthermore, various human-made Haiku with different qualities are also collected and mixed with AI-generated Haiku. The respondent have to guess the creator of those Haiku.

1.4 Outline

The paper is organised as follows.

- Section 1 delivers the motivation and the objective of research in automated poem generation.
- Section 2 of this paper explains necessary background information related to this topic. This section also covers recent related works.
- Section 3 explains the specification of this research. This section also lists the required resources to aid this research.
- Section 4 describes the detailed explanation of our implementation.
- Section 5 discusses evaluation method that will be used in this research.
- Section 6 provides the evaluation and analysis of the result.
- Section 7 concludes our research. This section also provides a discussion in relevant field.
- Finally, section 8 points several suggestions for the future development of this research.

Chapter 2

Background

2.1 Haiku

Haiku is a traditional Japanese poetry. Haiku is three-parts short poem consists of 17 *morae*, or 17 syllables in English Haiku. The syllables are further distributed so that the first and the last part have five syllables and the second part has seven syllables.

The problem occurs when Haiku is adopted into a different language, such as English [8]. The issue comes from different linguistic concepts and designs from Japanese to a foreign language. Some Japanese language concepts simply do not exist in English. Cultural difference is also one of the reasons of variety rules in foreign Haiku. Therefore, there is no strict rule on writing Haiku in English [9].

As stated before, Haiku consists of total 17 *morae*. However, *morae* is Japanese exclusive terms and does not exist in English. Therefore, people convert the definition of *morae* into syllables. However, this conversion causes controversy as *morae* and syllables are two different measurement units[10]. Thus, some people stated that 5,7,5 syllables distribution rule is preferred, but not mandatory [9]. In some conventions, Haiku writings do not have to strictly follow 5,7,5 syllables distribution, as long as it written in short-long-short sentence[11].

Another exclusive Japanese term in Haiku is *kireji*. Haiku consists of two different sub-ideas, separated by *kireji*, or ”cutting word”. In Japanese, *kireji* is used to extend emotional context. It may be used as punctuation. *Kireji* does not exist in English. As alternative, it is replaced by a punctuation mark, such as dash or question mark. In some cases, *kireji* is simply removed and understood as implied delay [12].

There is no specific genre in Haiku. Any subjects can be written in Haiku [9]. Haiku poem often delivers the content physically. This expression can be achieved

by choosing words that relate with one of the human senses such as visual, hearing, or touch. Moreover, Haiku uses *kigo*, or 'seasonal words' to refer and visualise the poem into one of the seasons in Japan. It is important to collaborate the five senses perception with correct seasonal theme [13]. However, modern Haiku may not contain any seasonal word. [9].

Below is the example of traditional Japanese Haiku with its translation:

Romaji:

*yuku haru **ya***
tori naki uo no
me wa namida
(Basho, tr. Shirane)

Translation:

The passing of spring –
The birds weep and in the eyes
Of fish there are tears.
(Translated by Keene) [14]

In the Haiku above, word *ya* serves as *kireji* or cutting word. It does not have an actual meaning, but rather to provide a slight delay or pause. In the translation, it simply translated as dash sign "–".

Haiku is simple and short which may be a good start on poetry generation research. Word choice is used to create unexpected yet creative phrases between words and lines which make this research challenging. We hypothesise that word vector may be able to capture such relation.

2.2 Automated Poetry Generation

Many experiments have been done in automated poetry generation. Those attempt varies in their approaches. Some of them were done by applying a genetic algorithm to construct a poetry[15], a chart parsing technique[5], or constraint programming approach[16]. Some of those approaches require wide arrays of resources such as lexical or grammar structure information. By using fixed grammar rules, the result-

ing outputs might be too rigid, while poetry often show flexible usage in grammar. Moreover, some of them have huge computational time because of the huge search space[5].

Rashel and Manurung develop template based poetry generation. The template is manually chosen from some famous poems by stripping out some words in certain Part-of-Speech tags, such as noun or verb. A new poem was created by filling in the stripped out words with a new one [17]. Colton et al. also developed similar template-based poetry generation[18]. In their approach, word slots are filled and paraphrased based on words similarity by the help of WordNet. The advantage of this approach is a seemingly less required resources compared to the rule or structure based poetry generation. However, an issue in this approach is that the system has no understanding of the meaning or the context of the generated poetry.

Several attempts were also made in specific Haiku generator. Wong et al. purposed a system that construct Haiku poems by combining lines from blogs. The output is chosen by selecting a poem with highest semantic similarity[19]. Another Haiku generation system was made by selecting two phrases obtained from a corpus. An additional phrase is then created to connect both phrases[20].

Specifically, we are interested in automated Haiku generator based on Words Association Norms system developed by Netzer et al.[21]. This system constructed Haiku based on predefined Haiku template and an input as the topic keyword. This approach is similar with our work. The core difference is that our work uses Word Vector model instead of Words Association Norms to compute word similarity. A disadvantage of Word Association Norms is the limitation of resource available, whereas Word Vector model can be trained with any raw corpus. Word Vector model is able to capture similarity of any new phrases or sentences by applying vector operation on the constructing words, where this cannot be achieved with Words Association Norms. Thus, their system is limited to a word as the input, where our system can take multiple words or phrases as the input. Finally Netzer et al., does not consider syllables count in their Haiku, while our system uses strict definition of Haiku which adds more constraints and challenges.

2.3 Poetry Measurement

One major issue in automated poetry generation is how to mathematically evaluate the result. This issue is challenging as poetry measurement is very subjective. Several

studies try to explore this matter and purpose concepts to objectively measure a poetry.

Manurung et al. mentioned that there are three aspects of a poem that can be scored. The first aspect of evaluation is the phonetic evaluation. This information can be evaluated through the poem's phonetic structure such as rhyme, metre, alliteration, etc. The second measurement is a linguistic evaluation. Word and syntax choices are measured at this point. The last part of the evaluation is semantics, where a poem is scored based on its semantics structure with respect to the desired semantics[6].

Colton et al. suggest a statistical approach in measuring a poetry. Evaluation is done by analysing each word and assign some scores based on several criteria. A flamboyance is defined by word frequency, where a word obtained less score based how frequent it used in the constructed poem. It also computes the relevance of each word from given baseline article. Lastly, appropriateness is a score defined by its average word's sentiment distance towards a given sentiment level [18].

2.4 Part-of-Speech Tagger

Part-of-Speech (PoS) Tagger is an application with a purpose to assigns part of speech label verb, noun, adjective, etc. to each input text word [22]. PoS tagger puts label based on certain tag set. This tag set usually has been modified so that it can express more specified and extended grammar such as verb-past, verb-present, adjective-superlative, etc[23]. Many different tagsets available for English. It varies in terms of their labels and their sizes[24].

Input

A frog jumps into the pond

↓

Output

A/DT frog/NN jumps/VBZ into/IN the/DT pond/NN

Figure 2.1: The Example of PoS Tagger

Figure 2.1 illustrates a use example of PoS tagger. An input sentence is tokenised and given a label for each token. The output of PoS tagger will be a list of paired token with its speech tag. Obtaining Part-of-Speech label is useful for further processing in natural language domain. One example is for word sense disambiguation[25].

2.5 N-gram Model

Given a sequence of data, n-gram is a continuous sequence of N tokens from the full data. Data might come in various types, such as text, signal, or numbers. In text data, tokens in n-gram can be a character, syllable, or word. For example, given a text "I am writing a Haiku". The 2-gram, or often called bigram of that text are (I,am) , (am,writing), (writing,a), (a,Haiku).

$$P(t_i|t_{i-1},...,t_1) \approx P(t_i|t_{i-1},...,t_{i-n+1}) \tag{2.1}$$

$$P(t_i|t_{i-1},...,t_{i-n+1}) = \frac{|(t_i,t_{i-1},...,t_{i-n+1})|}{|(t_{i-1},...,t_{i-n+1})|} \tag{2.2}$$

$$P(t) \approx \prod P(t_i|t_{i-1},...,t_{i-n+1}) \tag{2.3}$$

N-gram model uses n-gram to capture a probability of token t_i by observing last $n-1$ previous tokens, as shown in 2.1. The probability value can be computed by counting how many such n-gram sequence occurs in training data, divided by its last (n-1)-gram sequence total occurrence, as shown in 2.2. With chain rule as shown in 2.3, n-gram can approximate the joint probability of whole sequence.

N-gram model is simple and can be constructed easily. Despite its simplicity, n-gram model is proved to be useful in various language processing works, such as speech processing or language identification. N-gram model is also used in broader domains, such as heartbeat signal clustering[26] or an earthquake detection[27].

N-gram have several limitations. Firstly, the number of possible n-gram is very huge. This number increases with higher n. This issue leads to sparseness of the data. There are lot of n-gram sequence which has not occurring at all in the training data, thus making its probability zero. To handle this issue, smoothing technique is used. Several smoothing techniques that can be used are Add One smoothing or Good Turing. Secondly, n-gram assumes that the data only connected up to last n previous tokens. Therefore, n-gram model cannot capture any dependency in the text within distance more than n.

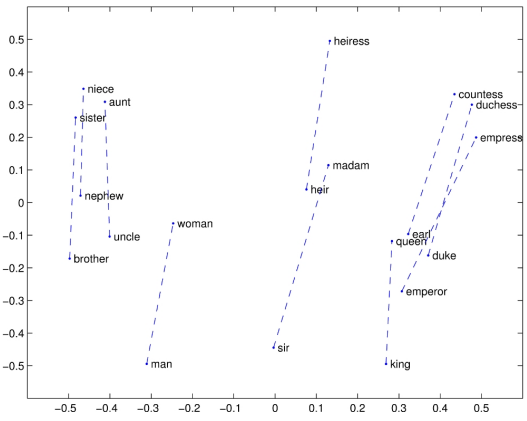
2.6 Word Vector Model

In word vector, words will be represented as vectors. Each word has its vector representation. Therefore, normal vector operation such as addition or subtraction might be applied to those words. There are several techniques that used to obtain those vector representations, such as using continuous bag-of-words and skip-gram[28] or using global word-word co-occurrence statistics[29].

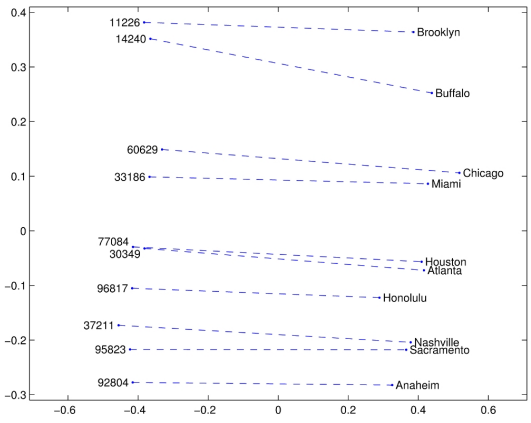
One simple usage of word vector is to measure the similarity of two words. As both words were represented as vectors, word similarity can be measured in standard vector distance measurements, such as Euclidean distance or cosine similarity[30]. For example, by using word vector representation we know that word ice, snow, cold, and winter are close to each other in terms of their vector.

Word vector can measure the similarity between phrases and sentences. This measurement is done by taking an addition of word vectors in each phrase and sentences to obtain two final vectors. The final distance of both vectors is measured to determine the phrase similarity[31].

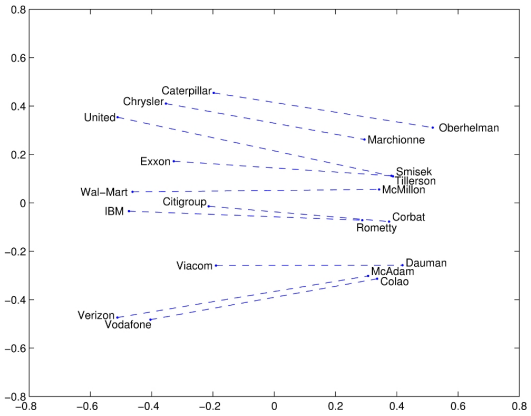
Word vector can capture linguistic regularities of words[32]. As shown in Figure 2.2, the vector representations tend to follow a similar relational pattern. Therefore, we can associate some words to obtain structural relations. Generally we can answer query of "a to b is like c to what?". Mathematically, we want to find D such that $a+b=c+D$. For example, we can answer "King to man is like a queen to a woman". In that query, we have to compute a vector that has closest distance with vector $vector(king)+vector(man)-vector(queen)$, which is $vector(woman)$. With word vector, we can capture semantic or linguistic relations of word only using a standard text corpus[30].



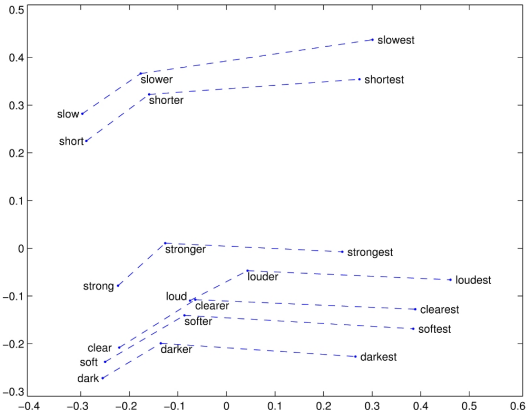
(a) Gender Relation



(b) City and Zip Code Relation



(c) Company and CEO Relation



(d) Comparative and Superlative Relation

Figure 2.2: Illustrations of Word Vector Model[30]

Chapter 3

Design

In this chapter, the design and specification of our system are explained. The required resources are also explained in this chapter.

3.1 System Specification

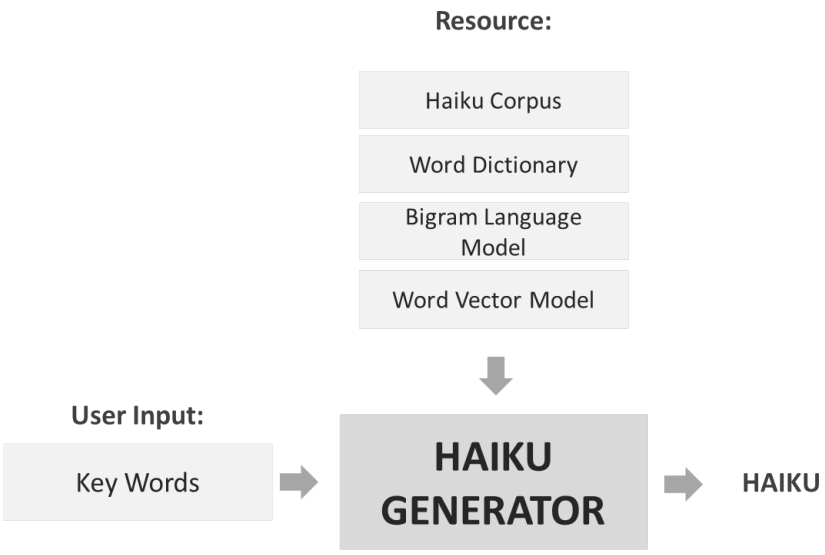


Figure 3.1: System Specification

Figure 3.1 above shows the specification of our system. This system requires resources of word vector, dictionary, and text corpus. This system also requires a user input of text keywords, which will be the topic of Haiku. In cases where users do not provide such keyword, a word will be chosen randomly from the dictionary list as a keyword. The output of this system is a random Haiku which relevant to the provided

keyword. This system generates Haiku randomly so that it has the capability to create different Haiku with exactly equivalent keyword and resources.

3.2 Resources

In this section, the required resources in our work are described. In general, our system requires four major resources: A dictionary, poetry corpus, Part-of-Speech tagger, and a text corpus. The main function of a dictionary is to list all possible words together with their required information. In our work, the required information is the syllable counts. Secondly, Haiku corpus and Part-of-Speech tagger are used to capture the grammar structure of the poetry without directly determine the grammar structure. Lastly, text corpus is used to extract statistical knowledge such as bigram word model and word vector model as tools to build the poetry.

3.2.1 CMU Pronunciation Word Dictionary

Word Dictionary is used as a knowledge base of English words. Carnegie Mellon University Pronouncing Dictionary (CMUPD) provides machine-readable pronunciation dictionary for over 133.000 words. It uses 39 set of phonemes based on ARPAbet symbol set. Each vowel contains lexical stress pattern information categorised into three different levels: 0 for no stress, 1 for primary stress, and 2 for secondary stress[33].

Word	Pronunciation	Syllables count
Haiku	HH AY1 - K UW0	2
Poetry	P OW1 - AH0 - T R IY0	3
Thesis	TH IY1 - S AH0 S0	2

Table 3.1: Examples of Syllabified CMU Pronunciation Dictionary

Bartlett et al. further improve the CMUPD and add additional syllables information. The syllabified CMUPD provides syllable boundaries that split between one syllable to another [34]. This information can be used to compute the total syllables of given word that is an invaluable resource for fitting the syllable constraints in Haiku. Table 3.1 shows some examples of words and their corresponding pronunciation data.

3.2.2 Haiku corpus

To capture the grammar pattern of Haiku, we have to collect Haiku as a corpus. This corpus is constructed by collecting Haiku from various sites, such as Poem Hunter, HaikuNow![35], HSA Haiku[36], frogpond journal[37], etc. The Haiku for this corpus is selected manually. As the purpose of this Haiku is to capture the grammar pattern, the collected Haiku do not have to follow strict 5,7,5 syllables rule. For this project, we successfully collected 43 Haiku. Thus, we have 129 Haiku lines in total.

3.2.3 OpenNLP

The Apache OpenNLP is a Java based toolkit for text processing. OpenNLP is used as main toolkit in this project. It provides various methods that help in NLP tasks in general such as tokenisation, parsing, or Part-of-Speech tagging[38]. Specifically, OpenNLP provides built-in maximum entropy approach of Part-of-Speech tagger[39]. This method is used to extract Part-of-Speech label in Haiku corpus.

3.2.4 Wikipedia Corpus

Wikipedia is an open encyclopedia where the data can be edited by anyone. Wikipedia is seen as a huge text resource as it consists of more than 4 millions articles available[40] in various languages. Wikipedia consists of more than 600 million English words based on a 2006 dump, and keep growing until now[41].

We use Wikipedia to construct a bigram language model. However, as Wikipedia size is considerably huge, we only capture bigram occurrences of words that exist in the dictionary as defined earlier.

3.2.5 GloVe Word Vector

Global Vectors for Word Representation (GloVe) is an algorithm developed in C by Stanford University to obtain vector representations of words. By using this code, we can obtain word vector representation of any language by using a standard unlabelled corpus text. Therefore, by using word vector representation, we try to indirectly understand words meaning and usage without using an additional resource such as WordNet or FrameNet, which might not available in some languages.

GloVe provides pre-trained word vectors constructed from various corpora. Those pre-trained data were obtained from various corpora, such as Wikipedia, Twitter, and

Common Crawl data. Those data also come with various vector dimension size from 50-D to 300-D. Those pre-trained word vectors are used directly for this project as they take considerably huge computational resources to construct them by ourselves.

Chapter 4

Implementation

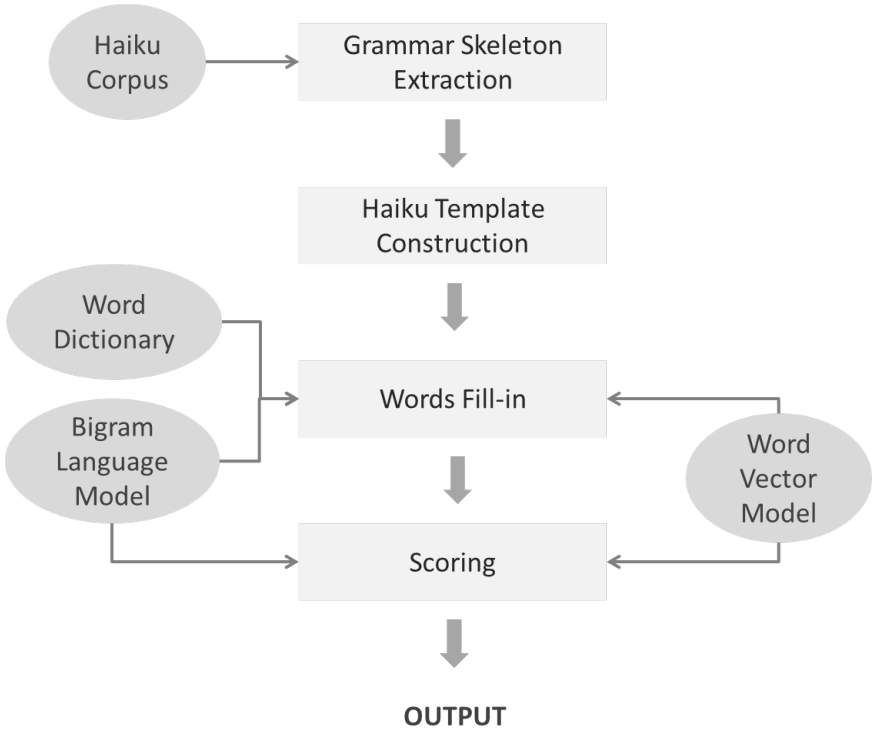


Figure 4.1: Algorithm Flow for Generating a Haiku

Figure 4.1 describes the general flow of our implementation. In the beginning, grammar skeletons are constructed from our Haiku corpus. Those skeletons are used to construct a new Haiku skeleton template. This template consists of several word slots with certain constraints such as syllables count or part-of-speech label. Words will be filled to construct a new Haiku. To avoid a long running time due to the huge space search, the system produces random several Haiku as samples, and pick one with the highest score with certain scoring method as an output.

4.1 Extracting Grammar Skeleton

The first step of our system is to extract grammar information of Haiku to capture their grammar pattern. To achieve this, we apply PoS tagger algorithm for every Haiku in Haiku corpus. For the next step, non-stop words are removed, thus leaving some parts of Haiku empty to be filled in later on. We called the output of this process as a **grammar skeleton**. An illustration of this process is provided in Table 4.1:

Original Haiku[35]	Grammar Skeleton
Indian summer	___/JJ ___/NN
mother dyes her graying hair	___/NN ___/NNS her/PRP ___/VBG ___/NN
the color of straw	the/DT ___/NN of/IN ___/NN

Table 4.1: The Example of an Extracted Grammar Skeleton of the Haiku

The reason of choosing this method over grammar tree or context-free grammar is because sometimes poetry does not follow standard grammar rules. By using formal grammar rule, we might achieve a less poetic result. By learning from existing Haiku, we hope that we can capture the unusual grammar structure used in Haiku or poetry in general which otherwise cannot be found on other texts[42].

4.2 Constructing Haiku Template

Based on grammar skeleton constructed in the previous step, we generate a new skeleton that serves as a template for the newly generated Haiku. The new grammar skeleton template is created by combining several skeleton patterns in the previous part.

Illustration of creating a new grammar skeleton based is shown above. Some base grammar skeletons are chosen as parents. The parents are chosen randomly as one approach in achieving randomness of the outcome.

The new grammar skeleton template is generated by applying crossover from the parents. In this example, each parent inherits one line of grammar pattern. In the implementation, the rule is not strictly forced the generation made by three parents. Therefore, more or fewer parents are possible.

Haiku in this project follow the rule of 5,7,5 syllables distribution. Therefore, for the next step the syllables count for each word are distributed. The word syllables count distribution are decided randomly. To achieve uniform distributions, we purpose

Skeleton 1
a/DT ___/NN ___/NN
under/IN the/DT ___/JJ ___/NN
___/NN ___/NNS
Skeleton 2
the/DT ___/JJ ___/NN
under/IN the/DT ___/JJ ___/NN
___/NN is/VBS ___/JJ
Skeleton 3
___/JJ ___/NN
___/NN ___/NNS her/PRP ___/VBG ___/NN
the/DT ___/NN of/IN ___/NN

↓

Generated Grammar Skeleton Template
___/JJ __/NN
under/IN the/DT ___/JJ ___/NN
from/IN the/DT ___/JJ ___/NN

Figure 4.2: The Example of a Grammar Skeleton Creation Process

a dynamic programming approach.

For each line, we compute how many different ways to construct a valid syllables count given the grammar skeleton. Let $syllable(w)$ returns the syllables count of word w, and $allSyllable(w)$ returns a set of all possible syllables count of word with the same grammar tag as w. The total different ways to construct a valid distribution up to $i-th$ word with a total of k syllables count are formulated as $f(i,k)$. $f(i,k)$ can be computed as below:

$$f(i,k) = \begin{cases} 1, & i = 0, k = 0 \\ \sum\limits_{j \in \{allSyllable(w_i)\}} f(i-1, k-j), & w_i \notin StopWords. \\ f(i-1, k-syllable(w_i)), & w_i \in StopWords. \end{cases} \tag{4.1}$$

The value of $f(i,k)$ is obtained by making sum over $f(i-1,k-j)$ where j is all possible syllable counts of the grammar tag of the $i-th$ word, except in a case where $i-th$ word is already defined. The total different ways to distribute the syllable counts in the first line of Haiku is provided in $f(N_1,5)$ where N_1 is the total words available in the first line. Similarly, the total different ways to distribute the syllable counts in the second and third line are provided in $f(N_2,7)$ and $f(N_3,5)$ respectively.

To randomly generate a valid syllable counts distribution, we run a backtracking process from the constructed table f. The syllable count of the $i-th$ word is picked by looking up the table $f(i,l)$. Assuming that we already picked the syllable counts for the word $j > i$, the value of l can be computed as $l = 5 - \sum syllable(w_j)$. Therefore, the probability of choosing x as the syllables count for the $i-th$ word is equal to $f(i-1,l-x)/f(i,l)$.

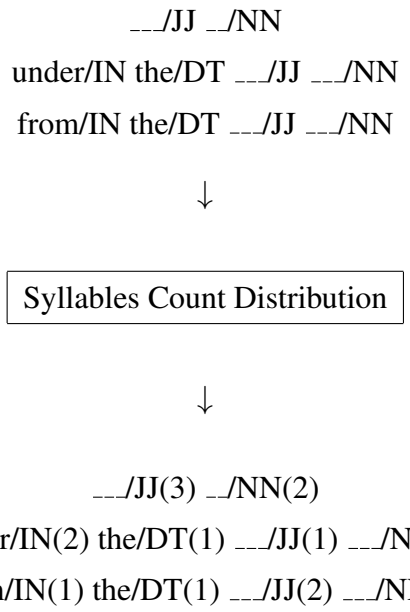


Figure 4.3: The Example of Syllables Count Distribution

Illustration in Figure 4.3 shown an example of the result of syllables count distribution. After applying the dynamic programming approach for uniform sampling in the grammar skeleton, we obtain additional information of syllables count for every word. This new information, as well as the grammar Part-of-Speech label, will be used in word filling.

4.3 Word Filling

In this phase, Haiku will be constructed with previously defined grammar skeleton as its basis. For each speech tag, we look up a word with an exactly same Part-of-Speech label and syllables count and fill it into the respective slot. Besides syllables count and Part-of-Speech constraints, a chosen word must follow additional connectivity and thematic constraint.

Connectivity constraint is defined as a constraint that avoid grammatically incorrect or invalid words to be chosen. Therefore, we want to avoid grammatically error phrases or unusual terms such as ”a ice”, ”she are”, or ”green cat” to occur in the resulting Haiku. In other words, the resulting Haiku must be valid with respect to a language model. In this work, bigram language model is used to capture the words connectivity.

Let assume that we already filled in the first *(i-1)* words, and have to fill in the $i-th$ word. Initially, all valid words with respect to syllables count and Part-of-Speech constraints are selected. From that set, all words that have no bigram occurrence with $(i-1)-th$ word are removed. Therefore, we have set of words that statistically valid to be selected together with the previous word. In a case where no such word exists, unigram language model is used instead.

the old neighbourhood
my cherry tree blossoming
on a stranger’s lawn[43]

A Haiku consists of three lines that may or may not be connected. As shown in the Haiku above, the first and the second lines are not connected. There is an invisible separator between lines. However, the second and the third lines are connected.

We implement a heuristic approach in our word filling to capture this characteristic. Let we want to fill in the word w_i where w_i is the first word of the second or the third line. Given a list of words with valid syllables count and Part-of-Speech constraints, we remove words that have no bigram occurrence with the previous word or with the start token. Therefore, in the case above, placing the word ”my” in the second line is valid because bigram occurrence of (START, my) exists. On the other hand, placing ”on” in the third line is valid because bigram occurrence of (blossoming, on) exists.

The last constraint is the thematic constraint. In this constraint, we want to choose words that relevant to the predefined topic. Word vector is utilised to capture the words relevancy. To apply this constraint, the word list obtained before is sorted based on

their vector distance to the topic. Then, we keep top-K words from the list. Finally, a word is randomly selected from the list and put it into the template slot. The words are picked randomly to assure that the system produces random and varied Haiku.

4.4 Post-Processing

Post-processing is the final phase in the Haiku construction. Post-processing is implemented to fix errors that occur in the previous step. It also used as a way to enhance the resulting Haiku. The analysis and causes of errors are not described in depth in this section as those discussions will be explored in the next chapter.

4.4.1 Cutting Word Placement

In the previous section, we define our heuristic approach on handling non-continuous sequence in Haiku. In this part, we utilise the non-continuous property of the Haiku to insert a cutting word. For two words sequence of w_{i-1} and w_i, if there is no bigram occurrence of (w_{i-1}, w_i) but there is bigram occurrence of (w_{i-1}, END) and $(START, w_i)$, a random cutting word is placed between them.

the old **neighbourhood–**
my cherry tree blossoming
on a stranger's lawn[43]

In the Haiku example above, bigram (neighbourhood,my) does not exist. Therefore, we can assume that the first and second line are not connected, and we can put a random cutting word at the end of the first line. In this example, we select a dash as a cutting word. In our implementation, the cutting word is chosen randomly from a choice of multiple dots (...), dash (–), coma (,), semicolon (;), or nothing. The reason we include nothing is that not every cutting words have to be written[12].

4.4.2 Article Inconsistency

In some cases, article inconsistency occurs in the resulting Haiku, such as ”a ice” or ”an great”. To fix this error, the determiner ”a” or ”an” are replaced with ”the” each time it paired with invalid words. We can remove both determiner ”a” and ”an” to ”the” without violating the syllables count in Haiku, because they have the same single syllable count.

4.5 Scoring and Ranking

In this part, generated haiku will be scored to determine its quality. We adopt the restricted definition of poetry by Manurung where a poem should be meaningful, grammatically correct, and poetic. As the haiku is generated based on previously defined grammar skeleton, we make an assumption that the grammar is already correct. Therefore, we can remove the second factor as a score. Accordingly, our scoring consists of two parts: meaningful and poetic. However, as poetic is very subjective, we alter the poetic measurement into a thematic measurement. The resulting Haiku has to be fit with the given topic.

$$M = \sum log\ P(i) \tag{4.2}$$

$$P(i) = max \begin{cases} p(w_i|w_{i-1}) \\ p(w_i|START) * p(END|w_{i-1})) \quad w_i\ is\ in\ the\ beginning\ of\ line \end{cases} \tag{4.3}$$

Meaningfulness score M is calculated by using the bigram probability. As shown in Equation 4.2 and 4.3, we define M as a sum of the log-bigram probability of each word in Haiku. As discussed before, some words at the beginning of lines might not be connected with the previous word. Therefore for special cases of words between lines, we pick the highest probability of joining word $i-1$ and i together or split them with a break.

$$T = \frac{1}{N} \sum distance(v_{topic}, v_{w_i})^2 \tag{4.4}$$

To measure the similarity T with the topic, we compute the distance of the topic's keyword and the Haiku in a vector space as shown in 4.4. Given the vector representation of the topic's keyword, we compute the average of the squared distance of each word in the Haiku. The average is used instead of the sum to avoid higher score for shorter Haiku. Cosine similarity distance is used in the distance function.

$$S = \alpha M + \beta T \tag{4.5}$$

The score will be a weighted sum of both components, as defined in the formula 4.5. Both α and β represent the weighting constant of each component respectively. Therefore, higher α means more 'meaningful' Haiku and higher β means more 'fit to topic' Haiku.

The system produces K random Haiku to be scored, and the system will pick one top scoring Haiku as the output. In this part, we also evaluate and define the value of K.

Chapter 5

Evaluation

The final phase of this research is to evaluate the system. In this phase, we want to measure our system by answering questions below:

- How relevant the generated Haiku with the defined topics?
- Do people like our Haiku?
- Can our system create Haiku that look like real human-made Haiku?

An online survey will be used to answer the questions above. In this survey, people will be given sets of Haiku on similar topics. For each set, several Haiku made both by human and AI and are mixed together. Finally, the respondent has to rate the Haiku based on several criteria. Additionally, a Turing test question will be asked to check whether our system can imitate a real human-made Haiku[44].

5.1 Survey Setup

The survey consists of 4 sets of Haiku, where each set has its specific topic. For each set, 4 Haiku will be provided based on specifications below:

- A random quality human-made Haiku

  This Haiku is used as a baseline for our measurement. By evaluating random quality Haiku, we can measure user's expectation towards the Haiku. A random quality human-made Haiku is collected from Reddit Haiku section, where any users can upload their Haiku.

- A good quality human-made Haiku

  This Haiku is used as an optimistic target of how good the evaluation of human-made Haiku is. By evaluating good quality Haiku, we can predict the required score for Haiku to be stated as a good one. This Haiku is collected from various winning Haiku in Haiku competitions.

- A random quality AI-made Haiku

  This Haiku is obtained by executing the Haiku generator once and collect the result directly without any further verification. We want to measure the expected result of our system, by using a random quality Haiku.

- A good quality AI-made Haiku

  This Haiku is used to measure the potential outcome of our system. This Haiku is obtained by executing the Haiku generator several times to collect several potentially good quality Haiku. Finally, one best Haiku is selected from the list of potentially good quality Haiku. To minimise subjectivity of the selection, a small team was created to select the Haiku. This team consists of five people in art and Japanese culture background.

As good quality Haiku are limited, four good quality Haiku are selected first. From those Haiku, topic keywords were extracted from each of the Haiku, resulting four keywords that will be used in this survey. The topic keywords in this survey are 'sadness', 'summer', 'sky', and 'war'. Based on those topics, AI-made Haiku are created. Finally, our Haiku used for this survey are:

- Keyword: Sadness

  - **Good Human-made Haiku (by: Janet R. Kirchheimer, WMF Haiku Contest 2012)**
    the ones who have died
    daughter of Shoah survivors
    she carries them all

  - **Random Human-made Haiku (by: *snoxxn*, Reddit)**
    it is refreshing
    yet sad to feel again that

which was forgotten

- **Good AI-made Haiku**
  a grave of people,
  nothing but mother to grieve
  with a deep sadness

- **Random AI-made Haiku**
  compulsion for death!
  communicating the turns
  the sadness is deep

- Keyword: Summer
  - **Good Human-made Haiku (by: Tom Painting, HaikuNow! Awards 2011)**
    Indian summer
    mother dyes her graying hair
    the color of straw

  - **Random Human-made Haiku (by: *FyreSign*, Reddit)**
    California sun
    brutal summer heating up
    get ready to burn

  - **Good AI-made Haiku**
    a hot afternoons...
    nothing but time to summer
    the play of sunshine

  - **Random AI-made Haiku**
    entire summer,
    in the enthusiasm

of the midnight club

- Keyword: Sky
  - **Good Human-made Haiku (by: Michael Henry Lee , HaikuNow! Awards 2012)**
    Leonid showers
    the sky continues falling
    one star at a time
  - **Random Human-made Haiku, by: *JoBear2484*, Reddit)** future streaks across
    the sky, a bright meteor
    past but dim shadow
  - **Good AI-made Haiku**
    oriental light,
    on the helicopter flight...
    with a distant earth
  - **Random AI-made Haiku**
    dancing the heavens
    the blinding on a blue night
    footage out the moon
- Keyword: War
  - **Good Human-made Haiku (by: Cherie Hunter Day ,HaikuNow! Awards 2010)**
    war memorial
    the shine on a bronze soldier
    from so many hands

– **Random Human-made Haiku (by: *SpeakWithThePen* ,Reddit)**
bloodied leaves descend;
how unsprung buds soon survive
their fallen fathers!

– **Good AI-made Haiku**
assault regiment;
i went to a major war,
into genocide

– **Random AI-made Haiku**
the operation
encouraging the allies...
future civil war

5.2 Survey Question

Initially, respondents have to answer several introductory questions. Those questions are asked to measure the general respondents' background and understanding about poetry and Haiku in general. In addition, English fluency level of respondents is asked.

In the next section, respondents will be given the Haiku above. For each Haiku, respondents have to provide their agreement with the statements. The respondents have to answer on a scale of Strongly Agree, Agree, Neutral, Disagree, or Strongly Disagree. The statements provided in this survey are:

- This Haiku Makes sense.
- This Haiku is beautiful.
- This Haiku is related to keyword *K*

There is also a question to ask whether the Haiku was created by human or computer. The respondents' respond are recorded to be used in the system's evaluation. Finally, analysis will be made based on the survey's result.

Chapter 6

Result and Analysis

6.1 Grammar Skeleton Extraction Result and Analysis

Poetry is very flexible in their grammar structure, including Haiku. Some Haiku do not follow formal grammar structures on its writing. Besides of an arguably artistic reason, we observe that the grammatical flexibility in Haiku appears because of the restriction of the syllables count.

strokes of lightning
one hit mountain frightening
dark clouds thunder loud[45]

Haiku must fit in strict 17 syllables. Because of this limitation, English Haiku writers tend to remove particles such as determiners. The writer may swap the word order to make the Haiku beautiful. The example of those Haiku is shown above. By using those Haiku as a corpus, we can capture unusual grammar patterns. Capturing this unusual grammar pattern might be very hard by using normal English rule.

the/DT bereaved/JJ girl/NN holds/NNS (holds should be a verb)

blades/NNS flying/VBG free/JJ —-/NNS (– should be a stop sign)

We notice that Part-of-Speech tagger is not perfect. It made some mistakes in tagging words, as shown in the example above. We cannot provide the exact accuracy of our grammar skeleton extraction in terms of Part-of-Speech label correctness, as the Haiku is unlabelled.

The grammar skeleton is our base to construct Haiku. Therefore, to minimise the mistakes, we manually analyse the results and make corrections on wrong labels. From 456 tokens, we corrected 14 wrong labels. Most mistakes are labelling a singular verb as a plural noun.

6.2 Word Fill-in Result and Analysis

Words filling relies on both word vector and bigram language model. A separate analysis of both components are presented in this section. The system seems able to fill-in Haiku template that relevant to the topics. It suggests that using word vector to predict word similarity works well, especially in this work. A comprehensive result regarding this claim is provided in the later section.

6.2.1 Word Vector Analysis

```
Words closest to "ice" =
    [ice, hockey, cream, skating, snow, rink,
    melting, frozen, chocolate, winter, arctic, polar,
    cold, melt, floe, icy, melted, water, glacial, glacier]

Words closest to "happy" =
    [happy, glad, feel, excited, delighted, good,
    pretty, wonderful, confident, feeling, satisfied,
    sad, happier, unhappy, disappointed, fun, thankful,
    hope, grateful, remember]
```

Word vector representation can capture word similarity and provides a list of similar words of a certain topic. Word vector can achieve this without understanding the actual meaning of the words. Above are some examples of closest words with some topics.

```
Words closest to "apple" =
    [apple, mac, cherry, juice, chips, fruit,
    blackberry, android, product, dell, cider, peach,
```

```
    pear, yahoo, chip, os, mango, hardware, banana, plum]
```

One issue appears as word vector has no capability in word disambiguation. When handling keywords that tend to have multiple meanings, word vector representations of those keywords are contaminated with all possible meaning. The word apple might be talking about a fruit or a technology. In the example above, first 20 words that have closest vector distance with word apple are mixed between both definitions.

a maker of taste,
I ate a mobile market
from a sugar plum

We cannot completely remove the ambiguity. In fact, if a user provides the system with a keyword 'apple', we do not know whether that user was talking about apple as a fruit or as a technology. A simple solution to this issue is to pick one definition and use it consistently. The issue occurs when the system is mixing both definitions and construct an arguably nonsense Haiku, as shown above.

To solve this problem, the keyword must be redefined. The keyword is modified by adding a word to construct an unambiguous phrase. For example, the keyword 'apple' should be modified into 'apple fruit' or 'apple technology'.

```
Words closest to "apple juice" =
    [apple, fruit, juice, mango, citrus, banana, peach, grape,
    tomato, pear, cherry, wine, vegetable, dessert, ripe,
    pineapple, bread, flavor, coffee, chocolate]

Words closest to "apple technology" =
    [apple, technology, tech, industry, digital,
    product, chip, hardware, electronics, chips, wireless,
    innovation, company, market, silicon, marketing, business,
    technological, maker, cisco]
```

By adding a vector 'fruit' or 'technology' to the original ambiguous vector 'apple', the system can obtain a new unambiguous vector. Examples above show closest words

to both apple fruit and apple technology. The result shows that the closest words are related and consistent for both keywords.

Another issue in an attempt to obtain word similarity by using word vector is that the definition of similar itself is not absolute. For example, we may agree with the statement of 'cat' and 'eagle' are not similar. In fact, a cat does not resemble an eagle at all. However, when we think globally, both cats and eagles are an animal, which a statement of 'cat' and 'eagle' similar is true as well.

```
Words closest to "winter" =
    [winter, summer, autumn, spring, weather, rainy, snow, season,
    cold, warm, fall, seasonal, alpine, ice, ski, weekend,
    skiing, harvest, year, warmer]
```

One another example is shown above. In terms of vector distance, 'winter' is very related to 'summer'. This case is true if we see them as a seasonal name. However, summer and winter are clearly different in physical condition point of view.

Similar with the previous issue, the solution of this problem is to put an additional keyword. Therefore, our general solution for both issues above is to mutate the provided keywords randomly. Given a keyword K, the system will pick a random keyword R from set of closest words to K. The new keyword $K' = K + R$ is now obtained.

6.2.2 Bigram Language Model Analysis

Bigram language model naively assumes that a word only depends in its exact previous word. Therefore, our system decision on *i-th* word is completely irrelevant with other words except the *(i-1) th* one. We feel this behaviour affects the resulting Haiku. Specifically, the produced Haiku feels less human-made like, less poetic and sometimes does not make sense. Our statement is very subjective. Therefore we took a survey and made an analysis on it to justify this. Further explanation is presented in the later section.

The sparseness of bigram model is another consideration. In a case where no possible bigram found while executing the word fill-in process, our system backed off to unigram. However, this step causes the system to create a Haiku with zero bigram probability. This issue further leads to negatively uncountable quality score due to

logarithmic computation. Although smoothing can be used to overcome this issue, the resulting Haiku still have a low-quality score that are very unlikely to be chosen.

Keyword	low quality Haiku	probability
sadness	155	0.62
summer	124	0.496
sky	135	0.54
war	123	0.492
cat	146	0.584
politics	138	0.552
love	147	0.588
mountain	141	0.564
evil	143	0.572
nature	133	0.532
average	**138.5**	**0.554**

Table 6.1: The Frequency of the System to Produce Low-Quality Haiku with Predefined Keywords

In a small experiment, we run the system to construct 250 random Haiku for each keyword. From those Haiku, we calculate how many times our system made low-quality Haiku due to bigram sparseness. In this experiment, we pick keywords arbitrary. Those keywords arguably represent common keywords of poetry. In Table 6.1, our system has 55.4% chance to make a low-quality Haiku due to bigram sparseness. It suggests that the system has to create 8 Haiku to produce at least one good quality Haiku with 99% confidence.

Table 6.2 provides similar experiment result. In this experiment, we let the system chooses the keyword randomly. In this complete random scenario, it shows that the system has a higher probability to construct a bad quality Haiku. The result suggests that keywords are an important aspect in the resulting Haiku quality. In the complete random scenario, the system has to construct 11 Haiku to produce at least one good quality Haiku with 99% confidence.

Those experiments suggest that bigram sparseness is unavoidable. Our suggestion to tackle this issue is to lower the weight of bigram probability score in the quality score computation. Therefore, it reduces the overall score penalty due to bigram sparseness.

Keyword	low quality Haiku	probability
unforeseen	168	0.672
tellingly	182	0.728
sharply	125	0.5
production	125	0.5
indicative	160	0.64
starlike	173	0.692
confined	146	0.584
calkin	182	0.728
tamarisk	173	0.692
flyer	156	0.624
average	**159**	**0.636**

Table 6.2: The Frequency of the System to Produce Low-Quality Haiku with Random Keywords

6.3 Scoring Result and Analysis

The scoring consists of two part, Haiku meaningfulness component and Haiku topic relativity component. The final score is a weighted sum of both components. Initially, we plan to adjust the weight by trial and error approach. However, an evaluation is needed each time a weight set (α, β) is used. Evaluating the result of the system is not an easy task and time consuming. Because of the limitation of time and resource, the weight factor is set to $(0.01, 1)$. The first reason for choosing this weight distribution is the difference in the value range of both measurements. Secondly, we set the meaningfulness score slightly lower to reduce the penalty due to bigram sparseness as discussed earlier. Moreover, we conclude that poetry might not have to use common terms. Some unique variations should be made to achieve a creative and beautiful result.

Our system randomly constructs K Haiku and selects one with the highest quality score as the output. To determine the value of K, we set up an experiment. In this experiment, several values of K are used. Then, we record the highest quality score for each K. As this experiment involves randomness, we run this experiment eight times and record the average quality score of each K.

Based on figure 6.1, the system tends to obtain a better quality scored Haiku with

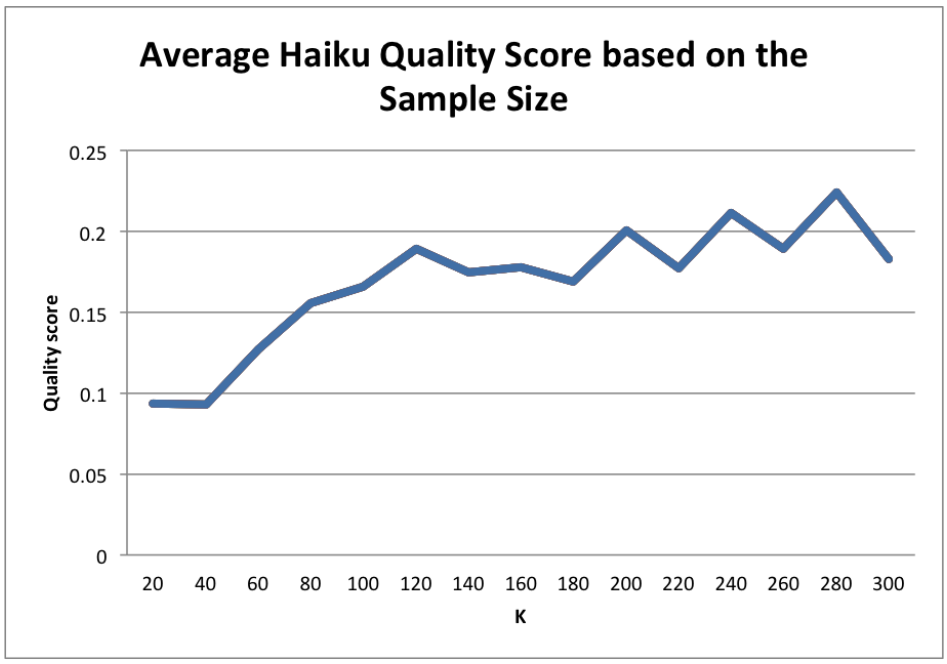


Figure 6.1: Average Haiku Quality Score Based on the Sample Size

higher K. However, the tradeoff of increasing K is the increasing computational time as more samples are collected. Based on experiment result above, we suggest picking K $\approx$ 120. With this value, the average quality score obtained is high by using a considerably small value of K.

6.4 Evaluation Result and Analysis

Evaluation was done by using a survey. Respondents are given a total of 16 poetry made by AI and by human. We analyse the respondents judgement toward the poetry in this section.

6.4.1 Respondent Overview

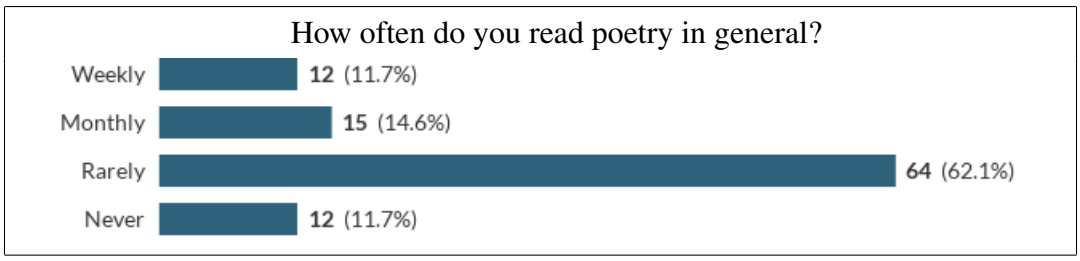


Figure 6.2: Distribution of Respondents' Familiarity in Poetry

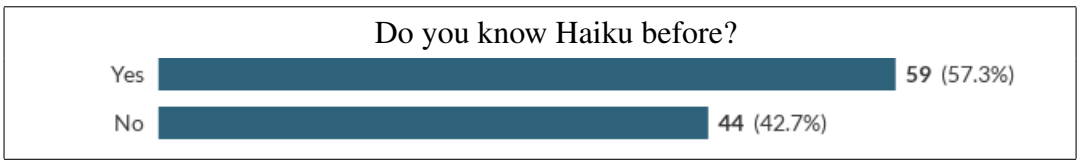


Figure 6.3: Distribution of Respondents' Familiarity in Haiku

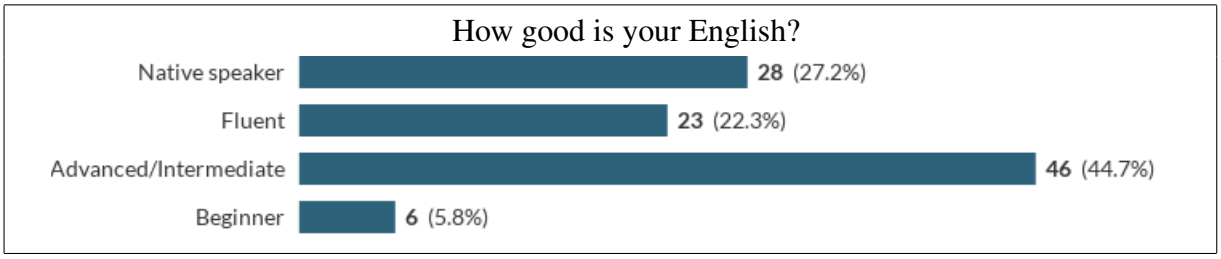


Figure 6.4: Distribution of Respondents' Competence in English

The survey was filled in by 103 respondents. Those respondents are classified by their familiarity with poetry and Haiku and by their English competency. The classification was done by using survey questions. The distributions of our respondents are shown in Figure 6.2, 6.3, and 6.4.

6.4.2 Meaningfulness Result and Analysis

To measure the meaningfulness of Haiku, respondents were asked whether they understand the meaning of the given Haiku or not. Respondents have to state their agreement by selecting one answer from strongly agree, agree, neutral, disagree, or strongly disagree. We assume that different topics might have different result. Therefore the results of this question for each topic set are provided.

Table 6.3 shows the raw results of this measurement. This table provides the frequency of the answers for each topic keyword. Figure 6.5 show the normal distributions of the respondents' agreement for each topic.

In both keywords of sadness and summer, the meaningfulness of good quality human-made Haiku are slightly lower compared to random quality one. Our hypothesis is that good quality Haiku in both topics talk about specific situation. Therefore, some users who have no knowledge or experience on that specific matters might think that the Haiku do not make sense.

The discussion of why some contest winner Haiku have lower meaningfulness score compared to random Haiku found in the internet can be very subjective. But

Result of statement "This Haiku makes sense"

Topic: Sadness

Quality	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Human-good	17	43	24	17	2
Human-random	41	37	16	8	1
AI-good	27	50	18	7	1
AI-random	1	22	38	40	2

Topic: Summer

Quality	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Human-good	25	46	15	16	1
Human-random	25	67	9	2	0
AI-good	11	42	21	25	4
AI-random	6	37	32	26	2

Topic: Sky

Quality	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Human-good	29	55	15	4	0
Human-random	11	51	27	13	1
AI-good	11	44	31	17	0
AI-random	5	21	32	38	7

Topic: War

Quality	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Human-good	17	62	16	8	0
Human-random	20	54	19	10	0
AI-good	14	56	23	9	1
AI-random	3	46	30	23	1

Table 6.3: Survey Result of Meaningfulness of Haiku

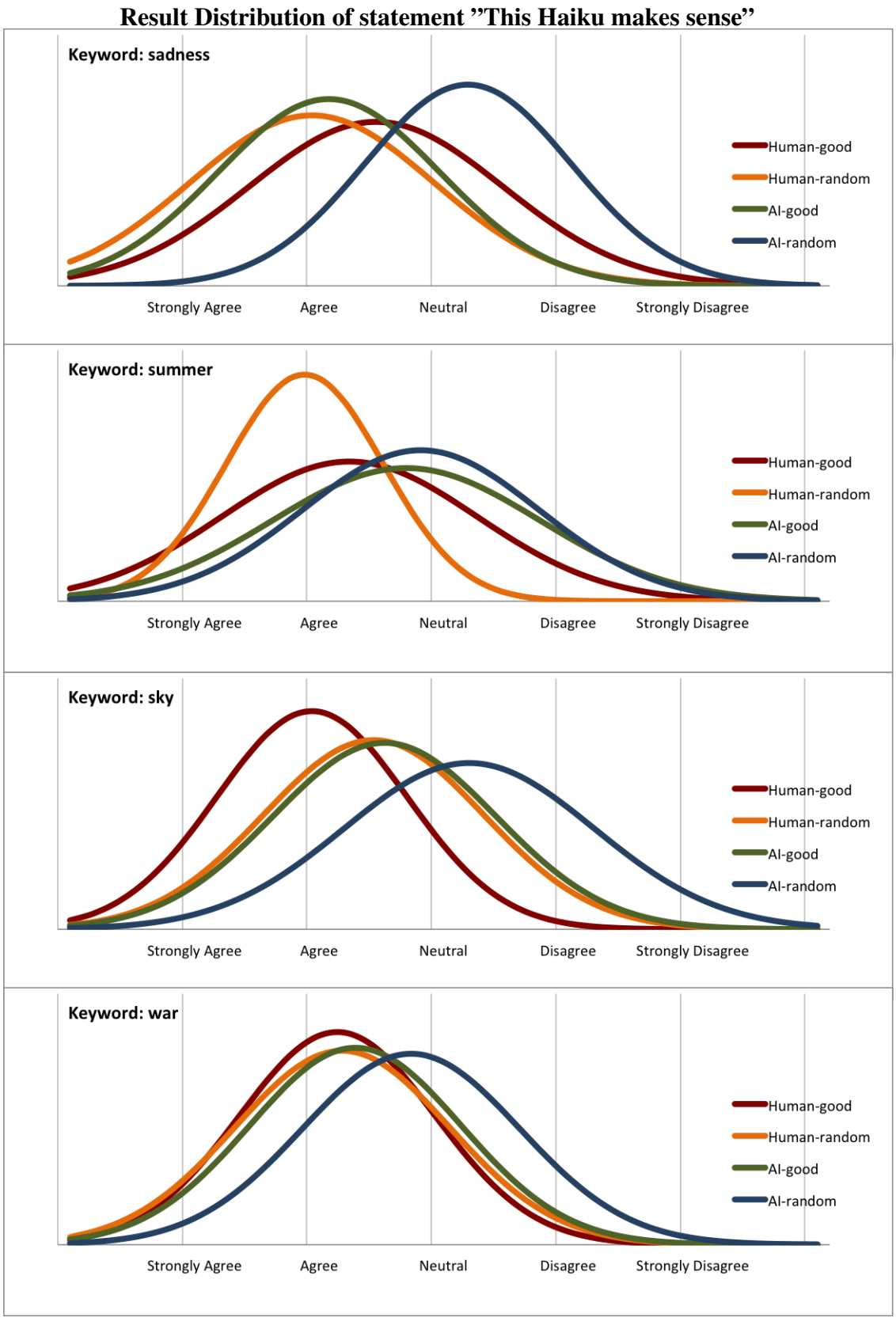


Figure 6.5: Normal Distributions of Meaningfulness of Haiku

from the general trend, Human-made Haiku have higher meaningfulness score in general compared to AI-made Haiku. However, our good quality AI Haiku have similar measurements compared to human-made Haiku in three out of four topics. This result shows that our system has a potential to create Haiku which is understandable and similar to human-made Haiku in sense of meaningfulness.

From all results, our random quality Haiku are the least make sense one. Random Quality Haiku have least average agreement of meaningfulness compared to all other qualities. As our Haiku meaningfulness is controlled by bigram language model, this result suggests that bigram language cannot capture the text's context in general. A possible reason of it is that bigram language model assume that words are not connected to each other besides its direct previous word.

6.4.3 Beautifulness Result and Analysis

Similar with the previous measurement, the respondents were asked about their opinion on Haiku's beauty. Respondents have to state their agreement by selecting one answer from strongly agree, agree, neutral, disagree, or strongly disagree.

There results are shown in Table 6.4 and Figure 6.6. In general, this result is similar with the previous measurement. Random quality AI-made Haiku are the least beautiful in average. Whereas good quality AI-made Haiku show a comparable result in terms of beautifulness with human-made Haiku.

From this result, it shows that the beautifulness order of Haiku is not the same as the meaningfulness order. In fact, good quality human-made Haiku show generally beautiful result even though some of them are less make sense. Therefore, we can assume that meaning does not strictly correlates with beauty.

Currently, our system has no control on the Haiku's beauty. The beauty factor of the resulting Haiku comes indirectly from the word usage which is controlled by the word vector model. Therefore, it might be a good idea to include another component that controls and measures the Haiku's beauty.

6.4.4 Topic Relatedness Result and Analysis

In this point, we want to measure the relatedness of our Haiku with the given topics. Similarly, respondents have to provide their agreement in one of the five scales. The results and their normal distributions are provided in Table 6.5 and Figure 6.7 respectively.

Result of statement "This Haiku is beautiful"

Topic: Sadness

Quality	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Human-good	11	37	35	17	3
Human-random	18	45	26	13	1
AI-good	14	43	28	15	3
AI-random	2	19	38	40	4

Topic: Summer

Quality	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Human-good	18	43	29	12	1
Human-random	4	27	42	27	3
AI-good	5	35	31	28	4
AI-random	3	25	32	31	12

Topic: Sky

Quality	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Human-good	29	47	24	3	0
Human-random	17	44	30	12	0
AI-good	5	33	40	23	2
AI-random	13	39	33	15	3

Topic: War

Quality	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Human-good	12	56	24	11	0
Human-random	27	49	17	9	1
AI-good	6	26	40	30	1
AI-random	1	13	46	40	3

Table 6.4: Survey Result of Beautifulness of Haiku

Result of statement "This Haiku is related to topic *T*"

Topic: Sadness

Quality	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Human-good	35	45	15	8	0
Human-random	29	50	18	6	0
AI-good	48	49	5	1	0
AI-random	17	52	25	8	1

Topic: Summer

Quality	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Human-good	20	42	20	15	6
Human-random	47	45	8	3	0
AI-good	47	48	7	1	0
AI-random	16	48	18	19	2

Topic: Sky

Quality	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Human-good	35	54	10	4	0
Human-random	20	53	20	10	0
AI-good	8	50	23	19	3
AI-random	16	54	19	13	1

Topic: War

Quality	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Human-good	27	62	9	5	0
Human-random	27	53	15	8	0
AI-good	40	56	7	0	0
AI-random	23	59	14	7	0

Table 6.5: Survey Result of Topic Relatedness of Haiku

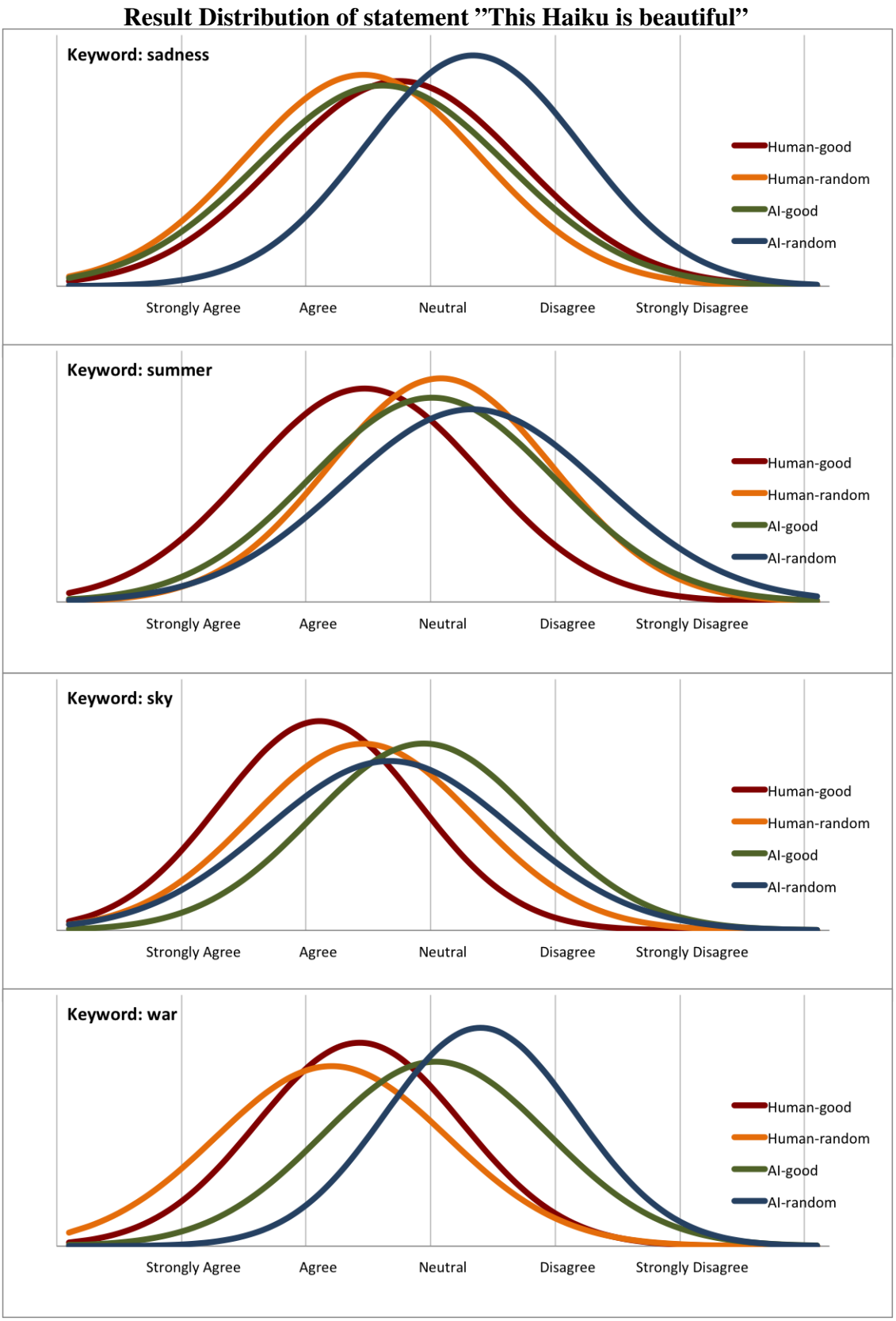


Figure 6.6: Normal Distributions of Beautifulness of Haiku

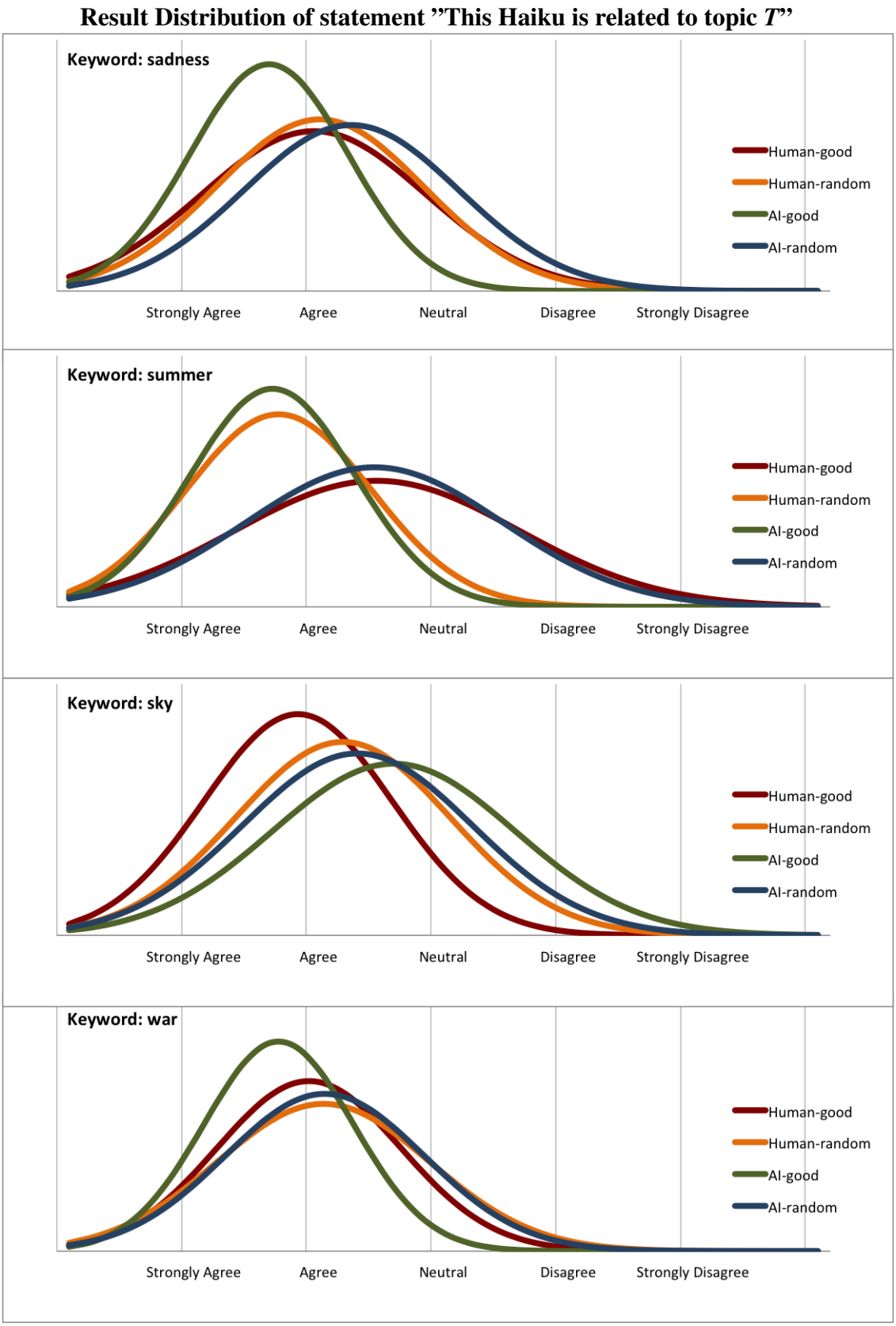


Figure 6.7: Normal Distributions of Topic Relatedness of Haiku

The result of this experiment shows that our system did a good job on making a topic specific Haiku. It shows that respondents are most likely agree that the Haiku is related to the topic. Additionally, there is no significant difference between the result produced by human-made Haiku and AI-made Haiku.

Based on this result, we conclude that our attempt on using word vector to construct Haiku with specific topic is successful. Word vector can be utilised to capture words that similar with the keyword, thus resulting Haiku that relevant to the topic.

6.4.5 Turing Test Result and Analysis

The final evaluation in this survey is a Turing test. In this test, respondents have to guess whether the Haiku was made by human or a computer. From this question, we analyse the result based on the majority vote.

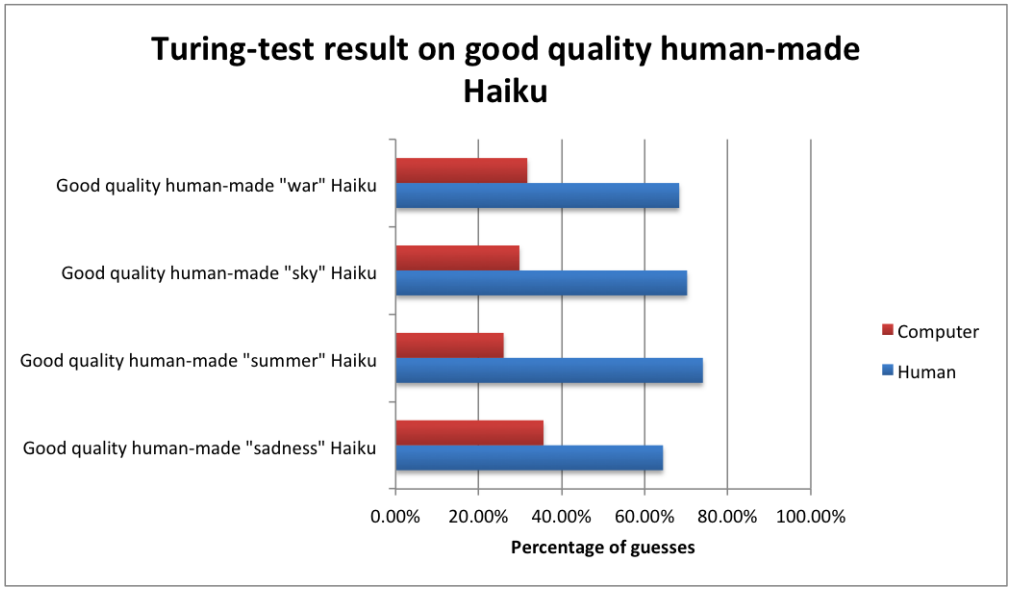


Figure 6.8: Turing Test Result on Good Quality Human-made Haiku

Figure 6.8 and Figure 6.9 provide the Turing test result on human-made Haiku. From those result, we can see that the majority of respondents correctly guess the writer for all Haiku. Therefore we can say that the respondents did a very good job on classifying the Haiku's writer.

From the test result shown in Figure 6.10 and Figure 6.11, the respondents fail to distinguish a good quality AI-made Haiku with sadness topic with 55.8% of respondents wrongly predicted the creator. Both good quality AI-made Haiku with topic of war and sky have close to balance distribution of guesses with less than 10% difference.

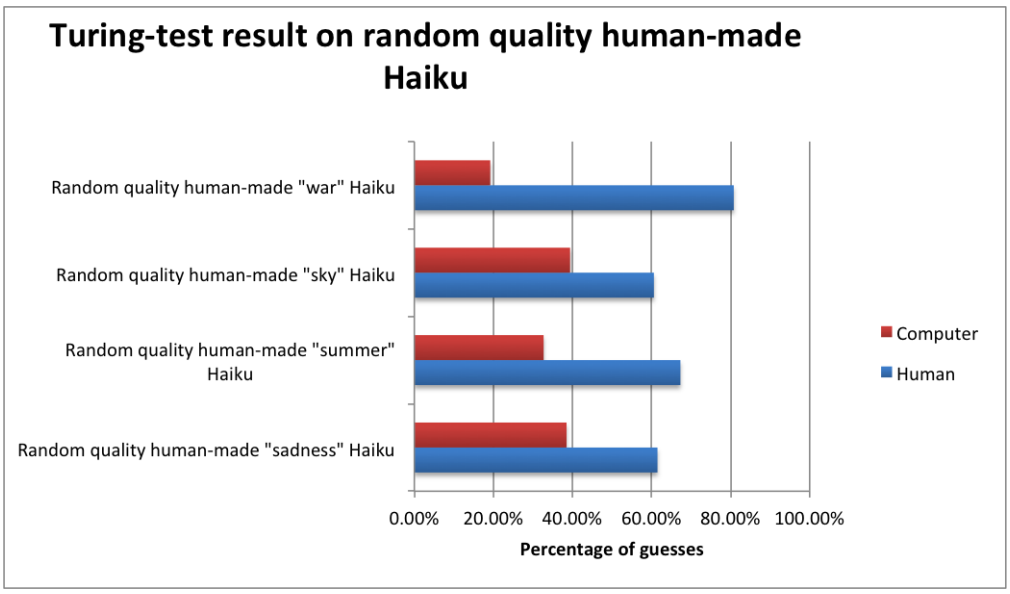


Figure 6.9: Turing Test Result on Random Quality Human-made Haiku

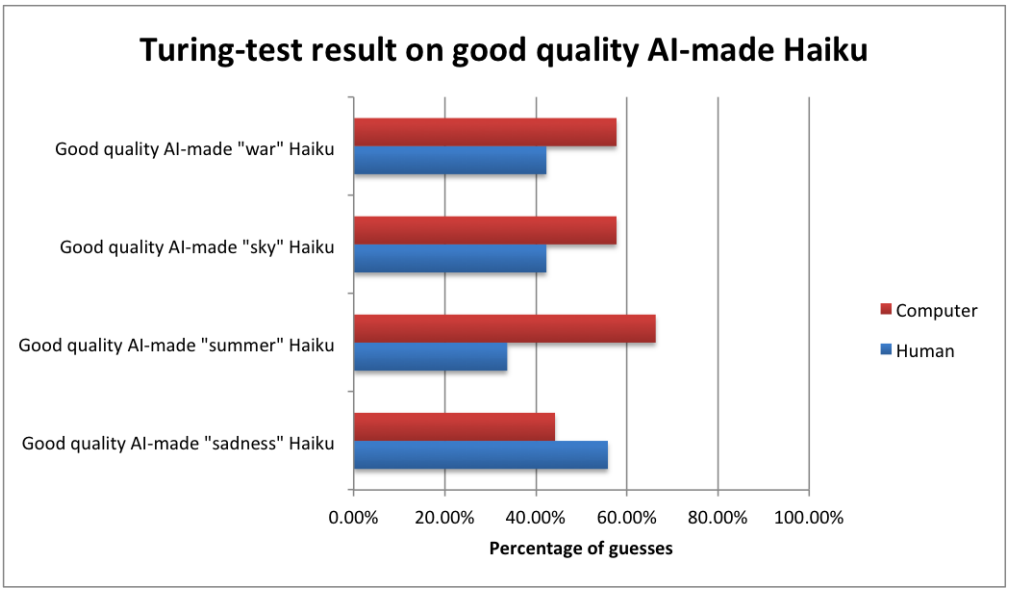


Figure 6.10: Turing Test Result on Good Quality AI-made Haiku

However, both of them are still classified correctly as AI-made Haiku. Respondents are able to correctly classify all random quality AI-made Haiku with high confidence level.

The Turing test results suggest that our generated Haiku are still easily distinguishable from real human-made Haiku. It shows a potential to imitates human as shown in the good quality result. However, in average cases, the system is unable to produce Haiku that mimics human writing style as shown in Figure 6.11. Therefore we

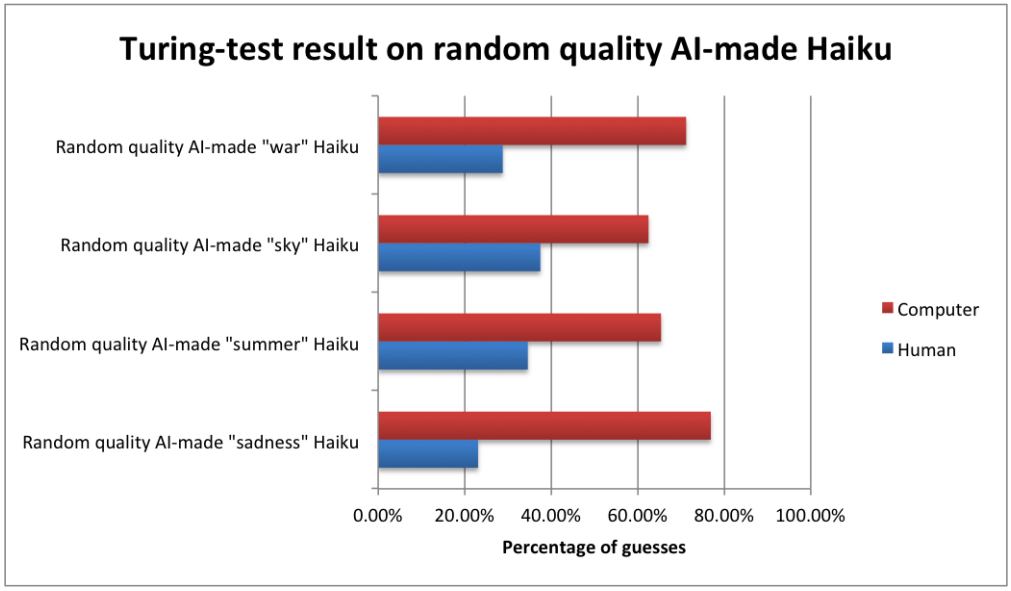


Figure 6.11: Turing Test Result on Random Quality AI-made Haiku

conclude that this system does not pass the Turing test.

Chapter 7

Conclusion

7.1 Conclusion

We designed an automated Haiku generator based on word vector language model. This system takes string keywords as an input. In cases where such input is not provided, the system will generate the keyword randomly. This system produces random Haiku, therefore the user will obtain different Haiku even with exactly equivalent input string. Our system constructs Haiku based on the provided keyword. This system construct multiple Haiku as samples. Those Haiku are scored and the highest scoring Haiku is selected as an output.

Our word vector model is pre-trained with Wikipedia corpus. With word vector model, the system is able to capture words similarity and relation without relying on additional resources. Furthermore, the system is able to create Haiku with a specific topic. Therefore, this statistic approach relies on unlabelled corpus except for Part-of-Speech tagging.

This system is further supported by Haiku corpus and Part-of-Speech tagger to capture the grammar pattern of Haiku. By using this corpus, we are able to capture unusual grammar structure of poetry which might be hard if using predefined grammar rules. Additionally, bigram language model is used to capture Haiku meaningfulness.

From the survey, our selected generated Haiku have considerably good result in terms of beauty and meaning.However, in general our Haiku are considered less poetic and less meaningful compared to human-made Haiku. One of the reason is that the bigram language model's independence assumption makes it hard to achieve continuity of our Haiku. However, our Haiku show a very good result in terms of topic closeness. Our system is able to construct Haiku that related to the given topic in general, shown

by the good average result in this survey. Additionally, our Haiku topic closeness result is shown to be similar with human-made Haiku topic closeness score.

In Turing test, respondents have to predict whether the Haiku was created by human or by a computer. In this test, respondents perfectly predicted the writer of human-made Haiku. One of eight AI-made Haiku was predicted wrongly by the respondents. In this particular Haiku, 55.8% respondents predicted the writer was a real human. Two of eight AI-made Haiku was correctly predicted with a percentage difference less than 10%. The rest of AI-made Haiku was correctly predicted by respondents with rather high confidence.

In general, this system is able to create Haiku with predefined keywords as a topic. However, in most cases, the resulting Haiku cannot compete with real human-made Haiku in terms of meaningfulness and beautifulness. The Turing test result also shows that users can easily recognises the AI-made Haiku.

WordNet is often used to detect word relationships[7]. However, WordNet capability might be limited to specific languages. It has no understanding on new or unusual phrases. In computational creativity, we believe that an understanding of new or creative phrases is needed. In the other side, word Vector is shown to be a very useful linguistic resource. It can be trained easily with a raw corpus and can capture word relationships. Word Vector is able to handle new phrases easily, by using vector operation on the constructing words.

We believe that Word Vector can be used for future computational creativity project. It is not bounded with language specific domain as long as a corpus on that particular language is provided. Therefore we hypothesise that Haiku generation on different language can be implemented by training the Word Vector on the desired language.

Chapter 8

Future Work

Some suggestions for further development of this system are:

- Currently, our system occasionally create a nonsense Haiku. We suggest that increasing bigram to higher order such as trigram might improve the system in terms of meaningfulness. An investigation on using different language model to capture meaningfulness of Haiku should be implemented.
- A plagiarism checking will be implemented in this system. This plagiarism checker is used to check whether our system accidentally construct a similar Haiku with another existing Haiku in the corpus.
- In scoring function, we only care about meaningfulness and topic closeness. Third measurement factor to compute the final quality score will be added. This score will capture the beautifulness of the Haiku. Several suggestions on this measurement are stress pattern, rhymes, sentiment analysis, or words usage.
- Theoretically, this system can be adopted in other languages or possibly other types of poetry simply by changing the resources into the desired language. Therefore, further investigation is needed to justify this claim.

Bibliography

[1] William Wordsworth. Preface to lyrical ballads (1802). *Romantic Prose and Poetry*, 1990.

[2] Hisar Manurung. An evolutionary algorithm approach to poetry generation. 2004.

[3] Simon Colton, Geraint A Wiggins, et al. Computational creativity: the final frontier? In *ECAI*, pages 21–26, 2012.

[4] Kim Binsted. Machine humour: An implemented model of puns. 1996.

[5] Berty C Tobing and Ruli Manurung. A chart generation system for topical metrical poetry. 2015.

[6] Hisar Manurung, Graeme Ritchie, and Henry Thompson. Towards a computational model of poetry generation. Technical report, The University of Edinburgh, 2000.

[7] Jane Morris and Graeme Hirst. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational linguistics*, 17(1):21–48, 1991.

[8] Kei Grieg Toyomasu. Haiku for people. URL `http://www.toyomasu.com/haiku/`.

[9] Haiku in english. URL `https://www.ryukoku.ac.jp/haiku/haiku_en.html`.

[10] RICHARD Gilbert and Judy Yoneoka. Haiku metrics and issues of emulation: New paradigms for japanese and english haiku form. *Language Issues: Journal of the Foreign Language Education Center*, 2000.

[11] Haiku Journal. What is haiku. URL `http://haikujournal.org/what-is-haiku/`.

[12] Ashley Capes. An introduction to haiku - form and structure: Kireji and kigo. URL http://ashleycapes.com/2013/08/26/an-introduction-to-haiku-form-structure-kireji-kigo/.

[13] Michael Dylan Welch. Becoming a haiku poet. URL http://www.haikuworld.org/begin/mdwelch.apr2003.html.

[14] Donald Keene. *Travelers of a hundred ages*. Columbia University Press, 1999.

[15] Ruli Manurung, Graeme Ritchie, and Henry Thompson. Using genetic algorithms to create meaningful poetic text. *Journal of Experimental & Theoretical Artificial Intelligence*, 24(1):43–64, 2012.

[16] Jukka M Toivanen, Matti Järvisalo, Hannu Toivonen, et al. Harnessing constraint programming for poetry composition. In *International Conference on Computational Creativity*, pages 160–167, 2013.

[17] Fam Rashel and Ruli Manurung. Pemuisi: a constraint satisfaction-based generator of topical indonesian poetry.

[18] Simon Colton, Jacob Goodwin, and Tony Veale. Full face poetry generation. In *Proceedings of the Third International Conference on Computational Creativity*, pages 95–102, 2012.

[19] M Tsan Wong, A Hon Wai Chun, Qing Li, SY Chen, and Anping Xu. Automatic haiku generation using vsm. In *WSEAS International Conference. Proceedings. Mathematics and Computers in Science and Engineering*, number 7. World Scientific and Engineering Academy and Society, 2008.

[20] Naoko Tosa, Hideto Obara, and Michihiko Minoh. Hitch haiku: An interactive supporting system for composing haiku poem. In *Entertainment Computing-ICEC 2008*, pages 209–216. Springer, 2009.

[21] Yael Netzer, David Gabay, Yoav Goldberg, and Michael Elhadad. Gaiku: Generating haiku with word associations norms. In *Proceedings of the Workshop on Computational Approaches to Linguistic Creativity*, pages 32–39. Association for Computational Linguistics, 2009.

[22] Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 173–180. Association for Computational Linguistics, 2003.

[23] Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330, 1993.

[24] Slav Petrov, Dipanjan Das, and Ryan McDonald. A universal part-of-speech tagset. *arXiv preprint arXiv:1104.2086*, 2011.

[25] Claudia Leacock and Martin Chodorow. Combining local context and wordnet similarity for word sense identification. *WordNet: An electronic lexical database*, 49(2):265–283, 1998.

[26] Yu-Chen Huang, Hanjun Lin, Yeh-Liang Hsu, and Jun-Lin Lin. Using n-gram analysis to cluster heartbeat signals. *BMC medical informatics and decision making*, 12(1):64, 2012.

[27] Alham F Aji, Petrus Mursanto, Setiadi Yazid, et al. Can smartphones be used to detect an earthquake? using a machine learning approach to identify an earthquake event. In *Systems Conference (SysCon), 2014 8th Annual IEEE*, pages 72–77. IEEE, 2014.

[28] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[29] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*, 12:1532–1543, 2014.

[30] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. URL `http://nlp.stanford.edu/projects/glove/`.

[31] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[32] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *HLT-NAACL*, pages 746–751, 2013.

[33] Carnegie Mellon University. The cmu pronouncing dictionary. URL http://www.speech.cs.cmu.edu/cgi-bin/cmudict.

[34] Susan Bartlett, Grzegorz Kondrak, and Colin Cherry. On the syllabification of phonemes. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 308–316. Association for Computational Linguistics, 2009.

[35] Tom Painting. Haikunow! 2011: Winners in the traditional haiku category. URL http://www.thehaikufoundation.org/haikunow-awards-for-2011/.

[36] Haiku Society of America. Harold g. henderson memorial award collection. URL http://www.hsa-haiku.org/hendersonawards/henderson.htm.

[37] Haiku Society of America. Frogpond. *Journal of the Haiku Society of America*, 2015.

[38] Jason Baldridge. The opennlp project. *URL: http://opennlp. apache. org/index. html,(accessed 2 February 2012)*, 2005.

[39] Alexandru Ceausu. Maximum entropy tiered tagging. In *Proceedings of the 11th ESSLLI student session*, pages 173–179. Citeseer, 2006.

[40] Wikipedia. Wikipedia's statistic. URL https://en.wikipedia.org/wiki/Special:Statistics.

[41] Samuel Reese, Gemma Boleda, Montse Cuadros, Lluís Padró, and German Rigau. Wikicorpus: A word-sense disambiguated multilingual wikipedia corpus. 2010.

[42] James Peter Thorne. Poetry, stylistics and imaginary grammars. *Journal of Linguistics*, 5(01):147–150, 1969.

[43] Marion Alice Poirier. Haikunow! 2011: Runner ups in the traditional haiku category. URL http://www.thehaikufoundation.org/haikunow-awards-for-2011/.

[44] Alan M Turing. Computing machinery and intelligence. *Mind*, pages 433–460, 1950.

[45] Floyd Floydson. Haiku thunder. URL `http://www.poemhunter.com/floy-dy-floyd-ra-floydson/stats/?poet=1173180&show=poem&poem=38540343`.