Automatic Haiku generation using vsm

Article · January 2008

CITATIONS
13

READS
374

3 authors, including:

Andy Hon Wai Chun
City University of Hong Kong
67 PUBLICATIONS **406** CITATIONS

SEE PROFILE

Automatic Haiku Generation Using VSM

Martin Tsan WONG & Andy Hon Wai CHUN
Department of Computer Science
City University of Hong Kong
Tat Chee Avenue, Kowloon Tong
Hong Kong SAR

Abstract: - In this paper, we present our current research progress in using an AI approach and modern Web 2.0 techniques to automatically create "modern haiku" poetry from text harvested from the blogosphere. Our haikus are generated using keyword seeds and the semantics of these keywords to extract lines from blogs to create a well-rounded poem. We believe this approach will yield much more human-understandable haiku poetry with much more realism than other traditional AI approaches as the language used in blogs is current as well as the events and cultural references. Our algorithm is based on formulating relationships between sentences as vectors using Vector Space Model (VSM) and then comparing relationships between sentences pairs using angle measurements based on simple cosine metric between the formed vectors.

Key-Words: - web2.0, Blog, Poetry Generation, Intelligence Systems

1 Introduction

Ever since Schank and Abelson introduced the world to Conceptual Dependency (CD) in the early 70's and exemplify by James Meehan's 1976 Yale thesis on a story-generation program called TaleSpin [17], researchers have been intrigued with the idea that a computer might one day be able to produce creative writing (story, poetry) automatically. This is not an easy task and requires research in many areas, such as natural language understanding, knowledge representations, common sense reasoning, etc.

Recently, Alex Dragulescu created a program called Blogbot [15] that automatically generates experimental graphic novels by harvesting text from web blogs. Based on the harvested text, a dynamic collage of images and text is generated using a keyword-matching algorithm. Inspired by this work, we propose a new approach to modern haiku poetry generation where lines of haiku are harvested from the blogosphere.

Modern haiku is a variation of tradition haiku poetry. Haiku poetry originated in Japan around the late 19th century. Nowadays, we can find haikus written in many different languages. In tradition English haiku, a poem consists of 5-7-5 syllable pattern of English words in three lines. A kigo (season word) is usually used. However, since the appearance of the "lily" and "bass" haikus by Nick Virgilio[12, 13], a new "modern" approach to haiku writing was created, which abandons the 5-7-5 pattern:

lily:
out of the water
out of itself

bass
picking bugs
off the moon

--Nick Virgilio (*Selected Haiku*, Burnt Lake Press/Black Moss Press, 1988)

Even today, people are still discussing whether traditional 5-7-5 syllable pattern should be followed. The Haiku Society of America [11], one of the biggest haiku societies in America, still holds debates on this issue. Even if there is no common consensus among haiku professionals on the structure of English haikus, there are some common practices of English haiku writing that all people follow:

1: Use of 3 lines of English words (or fewer) with no more than 17 syllables
2: Use of kigo
3: Use of caesure

The ability to automatically generate poetry that would be understandable for a human is not easy. Firstly, a lot of subtle knowledge about the world we live in and intricacies of human relationships and interactions are needed. Secondly, appropriate knowledge representations must be carefully selected so that we can adequately represent these subtleties in the English language. In addition, deep knowledge of word semantics is also important for the generation

process.

In previous AI research projects on automatic poetry generation, poetry contents are generated entirely by the computer from scratch. It may be guided by human-provided keywords, patterns, etc. or by a set of templates. In contrast, our approach uses a different way to generate haiku poetry. Instead of creating content from scratch, we make use of content harvested from the blogosphere. Blogs are popular nowadays and are used by millions of users daily. The content of our haiku will be selected from random blogs based on keyword semantics. We believe this approach will yield much more human-understandable haiku poetry with much more realism that those traditional approaches as the language used in blogs is current as well as events and cultural references.

2 Previous work on automatic poetry generation

Although we cannot find any research related to the software generation of "modern" haiku, there are several research papers related to poetry generation. For example, Pablo Gervás [3] implemented a case-based reasoning (CBR) poetry generation system, called ASPERA, to generate Spanish poetry. ASPERA generates poetry from different sources of information provided by the user:

1. A sentence (or set of sentences) in prose form to be converted into poetry,
2. A specific stanza that the final poem must comply with,
3. A set of cases for that specific stanza, and
4. A set of words that may be used in the final poem

Cases in ASPERA represent a corpus of verse examples. ASPERA was implemented using NASA's CLIPS rule-based system shell [3] and generates poetry in four steps typical of other CBR systems: **Retrieval**, **Reuse, Revise** and **Retain**.

Besides ASPERA, there is another system that generates Spanish poetry – COLIBRI, implemented by Belén Díaz-Agudo and Pablo Gervás [4]. COLIBRI also uses CBR and incorporates an application-independent ontology called CBROnto [4]. The poem generation process is similar to that of ASPERA. The main difference is the use of an ontology in COLIBRI that improves the inference power of the system as well as the representation and use of more explicit and general knowledge.

Besides Spanish poetry, Manurung, Graeme Ritchie [1, 2] implemented an English poetry generation system using an evolutionary algorithm. Manurung formulated the poem generation process as a state space search problem using stochastic hill-climbing search. The generation process is divided into two phases: evaluation and the evolution. During the "evaluation" phase, based on initial information, target semantics, and target phonetic, a set of evolutionary "individuals" are formed. This set of individuals will then be evaluated from different aspects, such as surface form, phonetic information and semantics. A score will then be assigned to each of the individual. In the "evolution" phase, individuals will be selected according to scores. The subset with higher scores will be selected for reproduction to produce mutated and, hopefully, better versions of the poems.

3 Our Proposed Approach

The proposed approach to story generation consists of the following steps:

1. keyword lexicon and line repository formation
2. haiku generation

3.1 Keyword Lexicon and Line Repository Formation

This is the pre-processing stage of our AI haiku generation algorithm. Two main components for haiku generation are formed – the keyword lexicon and line repository. Both will be used later in the second "haiku generation" step.

Keyword lexicon: The keyword lexicon is a database of keywords seeds used for both line repository formation and haiku generation. We started with the 500 most common words used in haiku writing as compiled by Professor Kenkichi Yamamoto[9, 14]. We further condense those 500 words to a small subset of around 50 keywords seeds that are more suitable for modern English haiku generation.

day	field	apple	cold	heat	phoenix	burning
year	color	sea	dream	sky	bell	sand
hot	garden	spring	rose	snow	bird	hunting
night	cool	winter	wind	rain	clothes	mountains
water	summer	dance	evening	warm	mirror	autumn
sun	voice	river	ice	moon	flower	mat
light	beach	tree	peak	leaves	swimming	dawn

Table 1: keyword seeds used by the system

Line repository: With the use of the Keyword lexicon created in the previous step, a line repository is formed. Line repository is the main storage of sentence fragments which will be used in the haiku generation process. The line repository formation process consists of 5 key stages:

1: Blog Entry Retrieval: Keyword seeds from our

Keyword lexicon are used as queries to the blog search engine. The returned XML datasets are analyzed and individual blog content retrieved.

2: HTML Parsing: Resulting blogs returned by blog search engine are then parsed

3: Sentence Extraction: Sentence(s) from the blog that contain the corresponding seed are extracted.

4: Segmentation: Extracted sentences are broken into fragments using common prepositions.

5: Filtering: Small fragments (less than or equal to 4 words) containing the keyword seed will be selected as a possible member of the Line Repository. Although there is no rule on number of words for modern haiku, one of the common practices is that there should not be more than 17 syllables total in the 3 haiku lines. In addition, basic syntactic knowledge will be applied to ensure sentence fragments can easily be "glued" back to form haiku.

In this paper, the blog search engine used is Technorati.

3.2 Haiku Generation

Revealing the Haiku moment

To be an interesting haiku, it must reveal the "haiku moment" – a Zen-like imagery. "Haiku moments were very much like other flashes of inspiration." [8] This is like a picture suddenly appearing in our mind as we read the haiku. In our modern haiku generation system, the first step is to create the general picture for the haiku moment. This is done by randomly picking three keywords seed from the formed keyword lexicon. The combination of selected keywords forms the general picture to create the haiku moment.

Sentences selection and Sentences matching

The selected keyword seeds will be used in each iteration of the haiku generation process as illustrated in the follow figure:

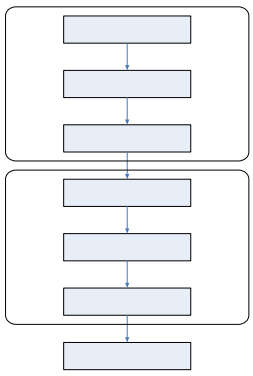


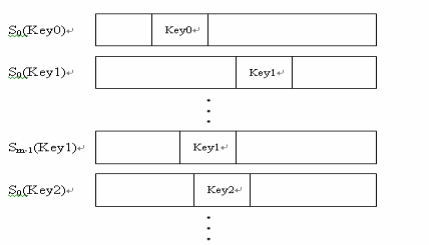
Fig 1: Each iteration of the haiku generation process

Sentence selection: Making use of the extracted keyword seeds as input, the corresponding sentences in the Line Repository is searched. After performing search, a set of sentences for each keyword are then found. Those sentences will then be fed into the second stage to perform the sentences matching process.

Sentence matching: The main purpose of this stage is to extract the useful information of each sentence, and then compare the semantic relationship between sentence pairs using VSM. The most semantically-related sentence pairs will then be chosen as the basic story plot of our haiku. In order to perform such a process, this system contains a keyword extraction engine, a knowledge space, a vector formation system, and a vector comparison system.

3.2.1 Sentence selection

In the sentence selection part, each selected keyword seeds will be used to find the corresponding sentence fragment containing the keyword. Except the keyword Key1, each keyword Keyi has to find m sentences containing the keyword Keyi. Similar to those keywords Keyi (where i $\neq$ 1), the keyword Key1 has to find only one sentence which containing the keyword Key1, rather than m. The reason for doing this will be explained in the following part. After the sentence selection system we have:



where S0(Key1) means the first sentence that containing keyword Key1

3.2.2 Sentence Matching

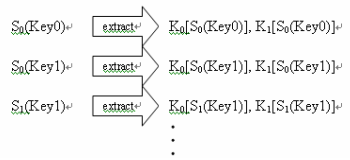
3.2.2.1 Sentence information extraction using keyword extraction

In this part, 2 keywords are extracted from each sentence, the extraction process is done with the use of TFIDF weighing scheme to evaluate how important a word is to a sentence.

For each selected sentence, we will calculate the TFIDF score for each word in the corresponding sentence. The TFIDF score is calculated using the following formula:

$$TFIDF(word) = termFreq(word) x \log\left(\frac{|corpus|}{DocFreq(word)}\right) \quad \text{.....(1)}$$

After the keyword extraction engine, we get 2 keywords for each sentence, and the corresponding symbol is shown as follows:

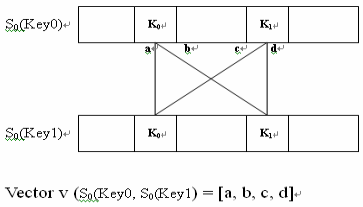


where K0[] is the first keyword extracted, and the item quoted inside the bracket is the original source (sentence).

3.2.2.2 Vectors formation

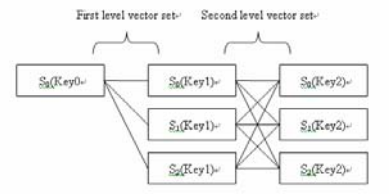
Vectors are used to describe the semantic relationship of a sentence pair. The pairs which have the most similar semantic relationship will be selected as a haiku output. Before explaining how to compare the characteristic between vectors, let's describe how vectors are formed.

Basically, each element in a vector of sentences pair is formed by providing a keyword from each sentence as a text-query to the search engine. In the case of our experiment, we use a common search engine – Yahoo! Corresponding matching results will then be assigned to the corresponding element in a vector. The number of elements in a vector depends on the number of keywords are extracted in the previous step – the value n. If there are n keywords in each sentence, then there would be n2 elements in a vector. In the haiku generation system, n=2 so there are totally 4 elements in a vector Figure shown below indicates how a vector is formed.



From the above figure, each keyword pair which is connected by a solid line will be the major element for raising text query to the search engine in turn. The values **a, b, c** and **d** are the matching results that are returned by the search engine. Such values will then be assigned to the corresponding elements in the vector. The reason for computing the vector in this way is to determine the likelihood that the second keyword would appear after the first keyword appears in the previous sentence.

The rule for the generation of vectors is: ***For each sentence S_i (i = 0,1,2….m-1) containing keyword Keyj+1, a vector is formed with the sentence S_k (k = 0,1,2….m-1) containing keyword Keyj , where j is called the vector set level.*** In each iteration, 2 levels of vectors will be generated.



The solid line connecting specific sentences represents the vector formed using the corresponding sentences. Example shown above is the case when ***m*** = 3, and there are totally 3 vectors in the first level and 9 vectors in the second level.

After generating all the necessary vectors, those vectors will then be submitted to the vectors comparison engine to find the most semantically-related pairs of sentences.

3.2.2.3 Vectors comparison

To compare whether a pair of vector is semantically related to each other, we make use of the traditional formula to measure the cosine of the angle between two vectors:

$$\cos\theta = \frac{v_1 \bullet v_2}{\|v_1\| \cdot \|v_2\|}$$

v_1 is a vector from the first level vectors, and v_2 is the one comes from the second level. The rule of finding corresponding vectors to perform the comparison process is: ***For each vector which is formed using sentences S_k (containing keyword Keyj) and S_i (containing keyword Keyj+1) in the first level, it should compare to all the offspring of S_i.***

The figure shown below indicates part of the actual vector comparison process in an iteration. All the lines connecting sentences are still representing vectors. There are 4 vectors **a, b, c** and **d** (vectors are represented using the solid line). As the vector **b, c** and **d** are offspring of S_0(Key1) (ie: the vector is formed using Sentence S_0(Key1) as one of the key factor for generating the corresponding vector), those vectors should be compared with vector **a** in turns. Thus in this state, a total of 3 vector comparisons will be made. A total of 9 vector comparisons will be preformed in this iteration. In general, if there are ***m*** sentences for each keyword ***Keyi***, then there should be m^2 comparisons in each iteration.

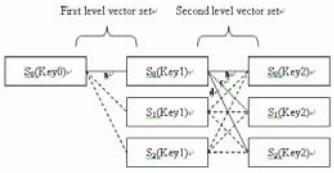


Fig 2: vector comparison

After the vectors comparison process, the vectors pair which has the value nearest to one (meaning that those vectors are not orthogonal to each other) will be selected, and the combination of those sentences will be a haiku output of the system.

Choosing a suitable value for ***n***: number of keywords to be extracted in a sentence, and ***m***: number of sentences selected for keywords Key1 and Key2 is important. There is no doubt that, large value ***n***, more information can be considered in a vector, but this also yields a large amount of intensive computation. Conversely, smaller the value n will cause the lack of information described in a vector. Similar case exists in the parameter ***m***, larger the value ***m,*** there will be more candidates to choose, but this also larger the number of vector comparison, as the number of vector comparison is mainly depends on the value ***m***.

4 Experiment Results

In our experiments, we extracted over 5,000 sentence fragments from over 10,000 blogs returned from Technorati. These fragments were then used to create the following poems. The following examples where generated using ***n*** = 2 and ***m*** = 3:

Generated haiku - Example 1:

Like mirror images
Of the moon falls
At river house restaurant

Keywords extracted from sentence 0: mirror, image
Keywords extracted from sentence 1: moon, fall
Keywords extracted from sentence 2: river, restaurant
cosθ= 0.95

Generated haiku - Example 2:

The snowy mountains
Search field
Of the honeymoon night

Keywords extracted from sentence 0: snowy, mountain
Keywords extracted from sentence 1: search, field
Keywords extracted from sentence 2: honeymoon, night
cosθ= 0.72

To compare our approach, we used the same calculation on a few human composed modern haikus. Below are some examples:

Awarded Haiku 1 (from Modern Haiku Magazine, Vol 37.3 by C.Avery)

They snap their fans
Open and closed
Day of the Dead

Keywords extracted from sentence 0: snap, fans
Keywords extracted from sentence 1: open, closed
Keywords extracted from sentence 2: day, dead
cosθ= 0.79

Awarded Haiku 2 (from Modern Haiku Magazine, Vol 37.3 by Greg Piko)

Another day summer
The butterfly, still safe
Under his steel pin

Keywords extracted from sentence 0: day, summer
Keywords extracted from sentence 1: butterfly, safe
Keywords extracted from sentence 2: steel, pin

cosθ= 0.85

Awarded Haiku 3 (from Modern Haiku Magazine, Vol 36.3 by Paul m)

Ringed moon
Rustle of the mouse
Near the trap

Keywords extracted from sentence 0: ringed, moon
Keywords extracted from sentence 1: rustle, mouse
Keywords extracted from sentence 2: near, trap

cosθ= 0.12

From the examples shown above, the cosθof human-generated modern haiku is around 0.7-0.8. When compared to those generated by the computer, they have similar cosθvalues (around 0.7-0.9). However, awarded haiku 3 has a cosθvalue that is quite different from other computer-generated and human-generated haikus. We believe the value of cosθis a good indicator for different haiku writing styles and/or techniques. Comparing computer-generated and human-generated haikus with similar cosθvalues shows they are similar in style, while the 3rd human-generated haiku is noticeably different in style. In modern haiku writing, there is not only one modern haiku writing technique in the literature. There are many different kinds of haiku writing techniques that writers nowadays follow [8].

5 Future Work

In this paper, we explored a new approach to modern haiku poetry generation using content from the blogosphere. Our implementation is still at its infancy and can be further improved and enhanced. First of all, our system analyzes the semantic connectedness of sentences within a poem. This allows our algorithm to following certain haiku writing techniques [8] that are based on semantics to create a more poetical and meaningful content. As future work, we plan to build an AI model to make use of these semantic measures as well as others to create a general way to evaluate certain qualities of poetry, both human-generated as well as computer-generated.

Secondly, we plant to extend our current model to consider not only single word nouns but also noun

phrases during the keyword extraction process. Noun phrases sometime contribute more on the meaning of the sentences than a single word alone. Text queries to Yahoo! can also be done with noun phrases as well for better accuracy.

Thirdly, we currently use Yahoo! as knowledge source and relationship measurements. For future work, we plan to extend the system to use multiple search engines, such as Google, Alta Vista etc. By integrating different features from different search engines, we can further enhance our common sense reasoning module. Besides search engines, there are other kinds of knowledge sources that we can tap into, such as Wikipedia.

6 Conclusion

In this paper, we explained our AI algorithm for modern haiku generation. It is based on the concept of Vector Space Model to model semantics and uses TFIDF to measure associations between words. The system also uses some basic knowledge of haiku structures and haiku keyword seeds to automatically extract content from blogs. These content fragments from a diverse collection of blog content are then "glue" together to form a meaningful poem using semantic measures. We believe this approach creates more interesting and human-like poems compared with traditional AI approaches that generate poems from scratch.

References:

[1] Manurung, H.M., Ritchie, G., and Thompson, H., Towards A Computational Model of Poetry Generation, In Proceedings of the AISB-00 Symposium on Creative and Cultural Aspects of AI. 2001

[2] Manurung, H.M., Ritchie, G., and Thompson, H., A Flexible Integrated Architecture For Generating Poetic Texts, Informatics Research Report EDI-INF-RR0016, University of Edinburgh. 2000

[3] Gervás, P., Automatic Generation of Poetry using a CBR Approach, In Proceedings of the 9th Conference of the Spanish Association for Artificial Intelligence (CAEPIA-TTIA 01). 2001

[4] Díaz-Agudo, B., Gervás, P., Pedro, A., and González-Calero., Poetry Generation in COLIBRI, In Proceedings of the 6th European Conference of Case Base Reasoning (ECCBR 2002).

[5] Church, K., Gale, W., Hanks, P., and Hindle, D., Using Statistics in Lexical Analysis, In Uri Zernik (ed.), Lexical Acquisition: Using On-Line Resources to Build a Lexicon. Lawrence Erlbaum Associates. 1991

[6] Fellbaum, C (ed)., WordNet: An Electronic Lexical Database. MIT Press. 1998

[7] Manning, C., and Schütze, H., Foundations of Statistical Natural Language Processing, MIT Press. 1999

[8] Reichhold, J., Haiku Techniques, 2000 issue of Frogpond, Journal of the Haiku Society of America (http://www.ahapoetry.com/haiartjr.htm). 2000

[9] Kondo, K.Y., Higginson, W.J (ed), The Five Hundred Essential Japanese Season Word (http://renku.home.att.net/500ESWd.html#SUMMER--THE%20SEASON)

[10] Modern Haiku Magazine, (www.modernhaiku.org)

[11] Haiku Society of America, (http://www.hsa-haiku.org)

[12] Haiku – Wikipedia, (http://en.wikipedia.org/wiki/Haiku)

[13] Nick Virgilio – Wikipedia, (http://en.wikipedia.org/wiki/Nick_Virgilio)

[14] Kenkichi Yamamoto – (http://renku.home.att.net/bios.html#KYamamoto)

[15] Blogbot : (http://www.sq.ro/blogbot.php)

[16] Meehan, J.. The Metanovel: Writing Stories by Computer, technical Report (YALE/DCS/tr074), Computer Science Department, Yale University. 1976

[17] Meehan, J, , TALE-SPIN an interactive program that writes story, In Proceedings of the 5th International Joint Conference on Artificial Intelligence (IJCAI). 1977

[18] Nguyen, D.P.T., Matsuo, Y., and Ishizuka, M., Relation Extraction from Wikipedia Using Subtree Mining, In Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence. 2007

[19] Liu, H., Singh, P, MAKEBELIEVE: Using Commonsense Knowledge to Generate Story, In Proceedings of the Eighteenth AAAI Conference on Artificial Intelligence