Context Independent Poem Recognition and Poetic Sentence Modification

Borui Wang
Stanford University
borui@stanford.edu

Huimin Li
Stanford University
huiminl@stanford.edu

Abstract

Recognizing the literature potential of articles can be particularly useful for text evaluation and generation. In this paper, we show that without knowing the length, format, and the context of the sentences, we are able to train a poem recognizer to tell whether a 3-sentence text excerpt is from a poem or non-poem article with a high precision of 90%. We discuss our training features, and we also propose strategies to modify non-poem texts to be more poetic based on the design and training results of our poem recognizer.

1 Related Work

Lyrics, poetries, public speech, and other artistic literatures are unique artifact of human natural language production, with the distinctive feature of having a strong unity between its content and its form. Tizhoosh et al. discussed the significance of poem recognition using various methods for text document classification [1]. They proposed several approaches to classify poem vs. non-poem by methods including using decision tree and naive bayes. Their classifiers were able to achieve above 90% accuracy overall, with the knowledge of the context information in the poems such as sentence format (e.g. shape and line break), article length, poem type (e.g. ballad, limerick and haiku), and the appearance frequencies of specific words. Their work also includes features covering rhyme and rhythm patterns, which reassured our thoughts that poem features like rhyme would be useful for recognizing poems.

Our approach differs from existing works, as we do not rely on context specific factors or patterns to recognize poems. Instead, we developed a context independent recognizer that only uses sentence-length independent features and we are able to recognize a wide range of poems of different types and styles. Our approach was inspired by the statistical analysis of the word-stress patterns in poems by Greene et al. [2], and we delve deeper into the impacts of using stress pattern as training features in the poem recognizer.

Automatic poetry generation has been regarded as a useful technological effort [3], but poetry-like sentence generation itself has been proved to be challenging as it requires intelligence, world and linguistic knowledge, and creativity [4]. Although there have been significant attempts to generate poetry and artistic languages in the past [5], very few research has been looking at making existing texts more lyrical and poetic. We think our proposed way of converting non-poem content into poetic ones based on what we learned while developing our poem recognizer, although naive and basic, can be a good starting point for this new type of poem generation.

2 Data Collection and Normalization

We need two main categories of text data - poems and non-poems. As we weren't able to find excellent corpus for poems, we crawled all the poems from poemhunter.com. The website contains over 10k poems of various style (classical, free verse, haiku, limericks, prose, etc) and mood categories. We made a crawler that collects all the poems' titles and contents in text format from their site. For non-poems, we wanted a source with great content control and moderation. We ended up choosing blog articles from Medium.com, a top-tier high quality blogging platform. We crawled the top 5000 feeds from around 100 topic collections ranging from Startup technology to fiction stories. We stored the posts' titles and contents in text format as well.

To streamline our training process, we concatenated all the lines of the poems into a single paragraph with proper addition of period and comma based on the existing punctuation and capitalization patterns in the poems. We also eliminated poems that are too short (less than 30 sentences) or mal-formatted after the concatenation. We then partition all poems into 10 sentence chunks, so that a poem of 50 sentences will be outputted as five partitioned files, each containing a 10-sentence partial of the poem. For Medium articles, similar to the way we process poems, we eliminated articles that are too long or mal-formatted, and we partition all articles into 10 sentence chunks. We ended up with having 16,452 10-sentence non-duplicated partial texts from poems, and 15,765 10-sentence non-duplicated partial texts from Medium articles for training and testing.

3 Feature Design and Implementation

We extensively experimented with features related to stress and rhyme structures as well as more straightforward features such as average word length, length of the sentence, and standard deviations of words. One important note is that since the sentences in poems have on average significantly shorter length than the sentences in Medium articles and poems have special line format and text structures, features around sentence length or text format will likely to give uninterestingly high precision in training result. In our feature design, we intentionally eliminated dependencies on sentence length and text structure so that we consider our feature engineering to be fair to reflect our linguistic insights. The text normalization step in our data collection process guarantees that we can't tell the difference using the text format in poems and Medium articles as all these texts are normalized, partitioned and properly concatenated into single lines. We ended up developing six features, where all of them except "Sentence Length (SL)" are independent of the length of the sentence.

Sentence Length (SL): As we noticed, the sentences in poems are usually shorter than those in non-poetic content; the length feature comes as the most straightforward one. The value of this feature is simply the number of words in all given sentences. The assumption is that the classifier will tend to classify shorter sentences as poems, and the longer ones as Medium articles, making this feature the least interesting one.

Ratio of Stress Levels (RSL): A lot of poems sound beautiful when they are read out aloud. This inspires us to explore the difference of pronunciation and stress patterns between poems and non-poetic content. We decided to use the CMU dictionary to extract the mappings from words to their pronunciations in the given phoneme set, which includes not only phonemes themselves, but also the stress for vowels. There are three levels of stress: 0 for no stress, 1 for primary stress (the strongest), and 2 for secondary stress. For example, the word "language" (L AE1 NG G W AH0 JH) has a strong stress on "AE", and no stress for 'AH'. We want to explore whether the stress levels would be proportionally different in the two types of text. For example, poems might have more primary stresses comparing to weak stresses due to its lyrical nature and the flow of pronunciation. To capture this characteristic, we designed this feature as a triple, with the ratio of primary stress count and secondary stress count, the ratio of no stress count and secondary stress count, and the ratio of primary stress count and no stress count. By calculating the ratio instead of the absolute counts of the stresses, we also eliminate the dependency on sentence length for this feature.

Stress Fluctuation (SF): Focusing on the pronunciation stress, we think one of the different patterns between poems and non-poems can be the variety of stress levels carried by the vowels. For multisyllabic words, usually we can find more than one stress level in the vowels, and frequently the primary stress vowels and weak/no stress vowels are interleaved with each other. Our first assumption is that poems tend to prefer words that have more fluctuations between lexical stress (i.e. stress on a given syllable) in itself, such as "mellowness" ('EH1', 'OW0', 'EH1') rather than "ripeness" ('AY1', 'EH1'). In addition, we think such fluctuation can be extended from lexical stress (i.e. stress on a given syllable) to stress across the entire text. Our second intuition is that poems tend to have more fluctuations among all syllables in the sentences. For example, in a poetic phrase from our poetry data set " in (IH0) winds (IH1) and (AH0) waves (EY1) of (AH0) sound (AW1)", the stress sequence is "0 1 0 1 0", which gives us a pattern of one fluctuation between every words. In this feature, we count the number of stress changes (from stress 0 to 1, and vice versa) across all syllables in the given text, and divide this number by the length of the sentences to get rid of this feature's dependency on sentence length.

Count of Most-shared Closing Stress (CMCS): A well-known trait of poems is the repeated rhymes at the end of each one or interleaving sentence. For example, from our poetry data set, in "Even then. It will still bear it all. Just so another will not have to fall " and "A sea gull overhead. In the distance he can see the land. He is alive and he's not dead, " we see the same phoneme with identical stress (AO1 for "all" and "fall", and EH1 for "overhead" and "dead") appeared in the last word of both sentences. Based on such observation, we designed this feature to return the amount of most frequently occurred primary and secondary stresses with the same phoneme location counted from the end of the last word of the sentences. For example, both overhead (OW1 V ER0 HH EH1 D) and dead (D EH1 D) have "EH1" at the second to last position of the words. The reason we are only counting the stress but not matching the "D" sound after the "EH1" stress in "overhead" and "dead" is because we saw many situations like "let" and "head" also can appear as rhymes in poems, so matching multiple phonemes would worsen our training result. Because we're extracting the same amount of sentences for poems and Medium articles and we only look at one word for each sentence, such feature is independent of the length of the sentence.

Standard deviation of Stress (SDS): This feature is based on the insight that because of the emphasis on rhymes, most poems will have a few dominate high/low stress phonemes and have a more skewed distribution of these phonemes than non-poetic content. We made a visualization (Figure 1) of the distributions of the primary and secondary stress phonemes from 500 pieces of 3-sentence-partials from poems and Medium articles and we were able to visually identify that poems have a more skewed distribution in its stress phonemes, which means certain stresses are used much more frequently than others in the chosen sentences (Figure 2). We therefore decided to return the standard deviation of the all the primary and secondary stresses of the sampled sentences in this feature to reflect such observation, which turned out to me really effective in identifying poems. This feature is independent of the length of the sentences as standard deviation identifies the degree of variations in data.

Average Word Length and Standard Deviation (AWL, SD): We observed that words used in poems are usually less explicit where longer words, especially long nouns such as "government", "technology", "employment" with explicit meanings are avoided. We therefore used the average word length in sentences as a feature. We also suspect that the standard deviation of word length would be greater in Medium articles dues to the use of longer words. Both features are independent of the length of the sentences.

Figure 1. Distribution of stress phonemes in poems (left) and Medium articles (right), where each color stripe represents the distribution of phonemes in a 3-sentences-partial.

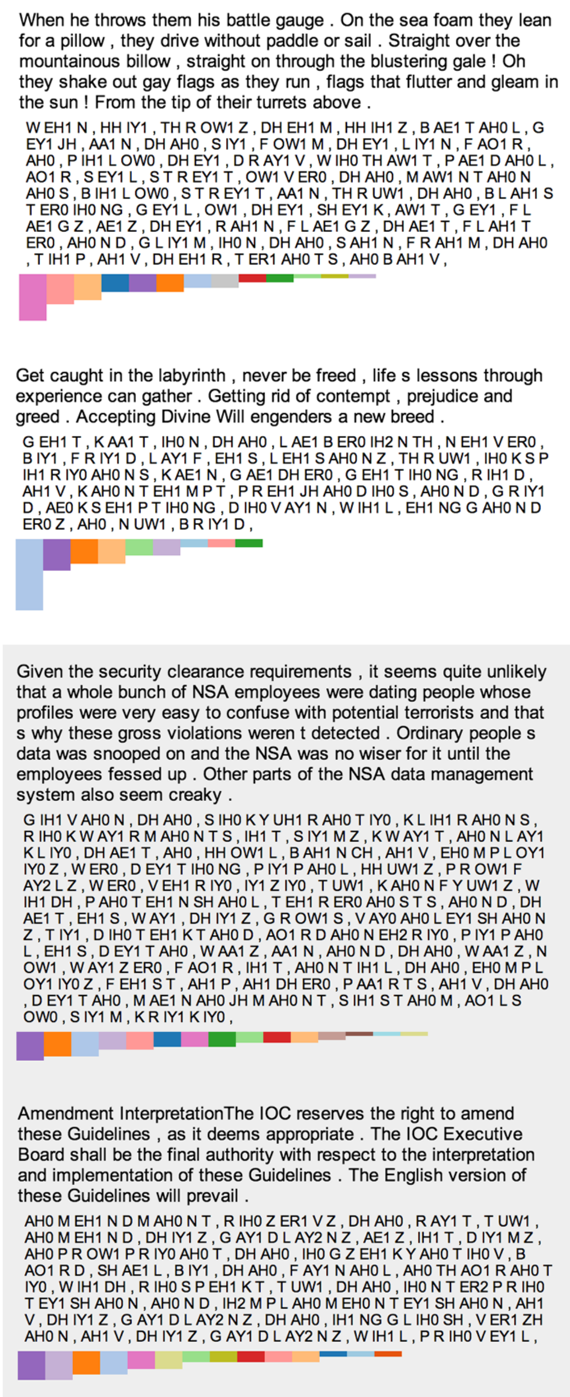

Figure 1. Sample distribution pattern of two poems (top) and 2 Medium articles (bottom) with original texts and pronunciations.

4 Training Result and Discussion

We used 90% of our data for training and 10% for testing. Table 1 shows the test set result with 4k poem training samples and 4k Medium article samples using all features above except the Sentence Length feature for training (RSL + SDS + CMCS + SF + AWL + SD), where we deterministically only choose 3 sentences from each sample that includes a non-duplicated 10 sentence partial from the original poem and Medium article.

3 sentences	Precision	Recall	F1
Poem	0.90	0.90	0.90
Medium article	0.90	0.90	0.90
Avg/total	0.90	0.90	0.90

Table 1

Our result illustrates that given 3 sentences from an arbitrary poem or Medium article, with only sentence length and format independent features, we are able to recognize whether the 3 sentences are from a poem or a Medium article with 90% precision. The high F1 score indicates that our features represent a general pattern seen in poetries disregard the type or mood of the poetry. Such result also confirms that our intuitions with the design of stress pattern related features are correct.

By investigating some misclassified poems and Medium articles, we found that a great amount of misclassified Medium article samples indeed have many shared features with poems. For these two examples:

“Personal tech was very present , and very , very visible among the again , self selected demo ists technology as a pose as much as a tool, Self augmentation , and an affinity for tech as simple , personal fantasy . Glass fit into this progression the next see me accessory , more than a see you tool”,

“I don't know , I'd admit . Fake geek girl , she said , letting the grin permeate the room . We popped various exotic pills that day , bought pop art , and walked directly into the stinging wind on our search for la Gait Lyrique .”

We see multiple closing rhymes in these Medium segments and several stresses to be more dominant than others. We think by adding basic topics models, our false positives in poems should decrease as very few poems talk about genetic technologies and startups. Since our feature wasn't distinguishing comma and question marks, and we didn't design features for word frequencies or word patterns in particular, we failed to identify poem samples that do not have a strong stress pattern like these:

"Lying on the floor surround me , surround me . Why'd you have to wait ? Where were you ? Where were you ? Just a little late , you found me , you found me ! The early morning , the city breaks . And I've been calling for years and years and years and years ",

"26 snowman's wedding she accepts her man for winter or worse . 27 software engineer's wedding she accepts her man softer or harder . 28 stringer's wedding she accepts her man for newsletter or worse"

This means that by designing features that exploit parallelism on the word level and pay attention to punctuations, we should be able to further decrease the amount of false negatives in poems. We also found that the misclassified poems have on average longer length than the average of all poems, while misclassified Medium articles have on average shorter length than the average of all Medium articles. However, all our features are designed to be length independent in mind. Such finding reveals that our other features might correlate with sentence length indirectly with a latent layer.

Table 2 shows the test set result of using 3 sentences deterministically chosen from each of the 4k poem and 4k Medium samples, but with a different combination of features (Sentence Length (SL), Ratio of Stress Levels (RSL), Stress Fluctuation (SF), Count of Most-shared Closing Stress (CMCS), Standard deviation of Stress (SDS),Average Word Length and Standard Deviation (AWL, SD)).

Features	Poem			Medium article		
	Prec.	Rec.	**F1**	Prec.	Rec.	**F1**
SL	0.91	0.94	**0.93**	0.94	0.91	**0.92**
RSL	0.70	0.64	**0.67**	0.67	0.72	**0.69**
SDS	0.86	0.82	**0.84**	0.83	0.86	**0.85**
CMCS	0.72	0.32	**0.44**	0.56	0.88	**0.69**
SF	0.76	0.65	**0.70**	0.69	0.80	**0.74**
AWL	0.80	0.68	**0.73**	0.72	0.83	**0.77**
SD	0.80	0.57	**0.67**	0.67	0.86	**0.75**
AW+SD	0.81	0.70	**0.75**	0.74	0.84	**0.79**
RSL+AWL+SD	0.79	0.75	**0.77**	0.76	0.80	**0.78**
SDS+CMCS	0.87	0.83	**0.85**	0.84	0.88	**0.86**
RSL+SDS+CMCS	0.87	0.86	**0.87**	0.87	0.87	**0.87**
RSL+SDS+SF	0.86	0.84	**0.85**	0.85	0.86	**0.85**
RSL+SDS+CMCS +SF	0.87	0.86	**0.87**	0.87	0.87	**0.87**
RSL+SDS+CMCS +SF+AWL	0.89	0.90	**0.89**	0.90	0.89	**0.89**
RSL+CMCS+SF	0.73	0.68	**0.70**	0.70	0.76	**0.73**
RSL+CMCS+SF+ AWL+SD	0.81	0.79	**0.80**	0.80	0.82	**0.81**
RSL+SDS+CMCS +SF+AWL+SD	0.90	0.90	**0.90**	0.90	0.90	**0.90**
RSL+SDS+CMCS +SF+AWL+SD+ SL	0.95	0.93	**0.94**	0.93	0.95	**0.94**

Table 2

Here we see that a single feature with Sentence Length (SL) gives a very F1 score as sentence length in poems (mean = 14.5 words) are usually significantly smaller than Medium articles (mean = 37.1 words), which as we mentioned, renders the sentence length feature less interesting. The Standard Deviation of Stress (SDS) is the one single most promising feature that gives a F1 score above 0.8 alone for both poems and Medium articles, which proves that our intuition of poems having higher standard deviations in stress was correct. Count of Most-shared Closing Stress (CMCS) performed poorly mostly because many non-classical poems in our training data do not have obvious closing rhymes in continuous sentence pairs or among interleaving sentences. We also tried using a Boolean instead of an integer value in this feature to simply check if duplicated stresses at the same phoneme location exist, but such alternative gave us the same result while we are extracting smaller set of sentences (3-4 from each poem/Medium article) and deteriorate as we are using more sentences because the chance of having a false positive in Medium articles get higher. Without SDS, we were also able to achieve an F1 score higher than 0.8 for both poem and Medium articles with the combination of RSL + CMCS + SF + AWL + SD. This shows that other alternatives to recognize poems do exist without considering the distribution of stresses. Stress Fluctuation (SF), Ratio of Stress Levels (RSL), Average Word Length and Standard Deviation (AWL, SD) all have moderate performance when used individually. But when combining with other features, they all help improve our training result incrementally, which indicate that they all have a different useful contribution to the F1 score. For example, in the poem sample "*I am a frightened girl they don't know I am. I am a sad girl who cries at night behind close doors. I am a distant girl who's out of sight form the public eye*" , using SDS or CMCS along would both classify this sample as Medium article, while adding RSL will correctly classify it as a poem, where RSL identifies the primary and secondary stress ratios that appear to be prominent in this sample while outweighing the contributions from SDS and CMCS.

5 Training with More Sentences and Data

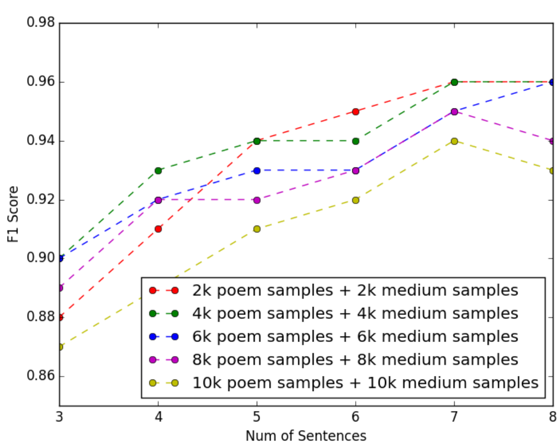

Figure 3

What if we were given more sentences (more than 3) to judge whether they're from poem or a Medium article? We also experimented varying the number of sentences we used from the poem and Medium article training samples, as well as changing training set size (Figure 3) As expected, we were able to achieve even higher precision given more sentences or more training data. As the number of sentences used for prediction increases, the prediction accuracy increases for all training set sizes. Given that our features are independent of sentence length and already effective in predicting poems or Medium articles with only 3 sentences, the features gain more power for distinction and are amplified as more sentences are seen for prediction.

However, we see a decrease in precision once we train with 7 sentences, which indicate the potential overfit for the training data. The other noticeable phenomenon is that the highest accuracy is achieved when the training set size is 4k (for both positive training example(poem) and negative ones(Medium articles)). When the training set is too small(e.g. 2k), and used with small number of sentences, the resulted accuracy starts at a relatively low level, however, if the training set is too large(e.g. 10k), the prediction accuracy drops significantly at 10k training samples, which also indicated a potential overfit for the training data.

6 Sentence Modification

If we think of the creative processes poets go through in making a piece of work, is there a way to model these processes by machines? In such process, not only strict restriction has been applied to the poem structure, but the poets must also demonstrate complex creativity at the same time. In the following sections of this paper, we present a model for semi-automatic generation of poetry by modifying medium articles to make them more poetic, while preserving original meanings. At a high-level overview, the modification procedure include the following steps:

(a) process the given text by sentence; get POS tag for each word in the sentence;
(b) obtain a list of similar words for each word based on wordnet synset and POS tag;
(c) for each word, replace it with one of its synonyms based on certain criteria, if no such synonyms can be found, use the original word;
(d) return the modified sentence.

There are several noteworthy details about the above procedure: in step (b), we manage to handle cases when the original word is in its special form, such as plural form for nouns, past tense for verbs, comparative form for adjectives, etc. In such cases, we transform the word into its lemma form first, find all synonyms, then add their corresponding forms to our similar word list; in step (c), we explored different criteria of picking synonyms based on the insights we learnt during classification phase. In particular, we focused on the directions of modifying based on rhyme, and modifying based on stress fluctuation. For rhyme-specific approach, we attempted the algorithm targeting at maximizing the count for one rhyme across the whole text - since we already know poems have distinct feature on rhymes, it is likely we can make a given text more poetic by changing words towards the dominating rhyme. (See pseudocode).

```
initiate global_rhyme_counter
for word w in input_sentences:
       initiate word_rhyme_set
       for synonym in similar_list(w):
              for each (position, vowel) in synonym_pronuncation:
                     word_rhyme_set.add((vowel, position))
       increment global_rhyme_counter for all keys in word_rhyme_set

Set highest_rhyme to be the key in global_rhyme_counter with highest count

modified_sentence = []
for word w in input_sentences:
       if w has synonym:
              add the synonym that contains highest_rhyme to modified_sentence
       otherwise:
              add the original word w to modified_sentence
return modified_sentence
```

In this way, we took into consideration all possible rhymes for all synonyms, and thus it is guaranteed to reach the maximal rhyme in the end. For stress-specific approach, we mainly focused on the stress fluctuation count with a greedy algorithm. The goal is to pick synonyms such that the global stress fluctuation can reach the largest. In our approximate algorithm, we consider the synonym that can maximize the stress fluctuation within each bigrams. (See pseudocode).

```
modified_sentence = []
prev_word = None

for word w in input_sentences:
       initiate fluctuation_counter
       for synonym in similar_list(w):
              set value to fluctuation_counter[(prev_word, synonym)]
       pick the synonym that gives highest fluctuation for the bigram
       if picked successfully:
              add the synonym that contains highest_rhyme to modified_sentence
       otherwise:
              add the original word w to modified_sentence
return modified_sentence
```

Finally, we evaluate our modification from two sides: from the quantitative side, we pipe the modified text back into the classification model we trained, and evaluate how many of the modified text are classified as poems now; in addition, we also look at our modified text to gain a qualitative perspective.

7 Modification Result and Discussion

By changing our modification algorithm, we obtained very different result in terms of the success rate, i.e. how many modified Medium articles can be classified as poems. We use a testing set of 500 Medium articles. When using modification based on rhyme, we only successfully transformed 2 Medium articles into poems, and our initial fluctuation count approach (trying to maximize the total number of fluctuation) did not perform well either with only 2 modified to poems given the previous classifier we built. In addition, the ratio between successful modification and total modification (based on whether the classification label has changed) is very low. To understand why our model did not perform as expected, we took a deeper look into the sentences before and after modification. For the first model (rhyme), the effect is observable, for example, for sentence,

"At others, downright scary. But sometimes it has also felt absolutely exhilarating. "

has been modified into,

"At others , downright chilling . But sometimes it has also felt absolutely elating."

where the rhyme IH0 is now dominant in the modified text. However, we suspect there are two problems associated with this approach. First of all, the number of words get modified is relatively small. For each Medium article (i.e. three sentences), usually only two or three synonyms can get selected since we require the synonym to have exact ending rhyme as the one with highest count. This, although increases the maximum rhyme count, is not sufficient for the classification result to change. The second suspicion is that although our algorithm guaranteed optimal solution for the highest rhyme count, it does not necessarily contribute enough to the standard deviation, which is used as a feature in our classifier. For the fluctuation-based approach, we noticed while picking synonyms, the algorithm always favors longer words/phrases, as it may possibly introduce more stress change. This contradicts to the intuition that poems are usually concise and short. As a matter of fact, the poetry might actually lean towards lower fluctuation counts due to its conciseness. Inspired by this, we flipped our algorithm to pick the synonyms that give the lowest score of fluctuation for each bigram. Furthermore, we add logic to favor the shorter words/phrases whenever there is a tie. To our surprise, it gives us 332 successful modifications out of 500 test examples. In addition, the ratio between success modification and total modification is extremely high, which reaffirms our hypothesis (Table 3).

Feature to focus	Total modified labels	Successfully modified to poem
Rhyme	13	2
Fluctuation count(max)	13	2
Fluctuation count(min)	333	332

Table 3

8 Conclusion

We showed that by only looking at stress patterns and non-context specific features of excerpts as small as 3 sentences, we can recognize if such text is poetry or not with high precision, which means that knowing other specific rhyme structures, formatting of the text, punctuations, length of each sentence or the full text are not necessary for recognize poems. We also showed a technique to create poetic sentences according our findings by modifying non-poems. A significant limitation in our design is our definition of poem, which ranges from haikus to limericks in our training data, and we can't identify poems of specific type, but rather poetic content in general. However, this also indicates that the stress related features represent a general pattern seen in lyrical and poetic texts. We think our work can be more fine-tuned toward analyzing certain type of poetry so that certain features can be distilled to have more impact on the specific type of text. We also think our sentence modification technique can be much improved if considering modifying the syntax structures of the original sentence or extracting semantic meanings for matching with better alternative expressions. In terms of potential applications, we think our work would allow identification and comparison of text in terms of their poetic potential for content extraction and editing, especially for movie scripts or lyrics. Our sentence modification technique can be used in text preparation processes for suggesting edits or producing more catchy phrases for commercial purposes.

Acknowledgments

We used the Scikit and NLTK libraries for python to develop our training scripts. And we used the Cmudict and Wordnet corpus for analyzing and modifying sentences. We'd also like to thank the StackOverflow community for helping us to debug our code.

References

[1] Tizhoosh, Hamid R., and Rozita A. Dara. "On poem recognition." Pattern analysis and applications 9.4 (2006): 325-338.

[2] Greene, E., Bodrumlu, T., & Knight, K. (2010, October). Automatic analysis of rhythmic poetry with applications to generation and translation. In Proceedings of the 2010 conference on empirical methods in natural language processing(pp. 524-533). Association for Computational Linguistics.

[3] Milic, L. T. (1970). The Possible Usefulness of Poetry Generation.

[4] Manurung, H. (2004). An evolutionary algorithm approach to poetry generation. Chicago

[5] Colton, S., Goodwin, J., & Veale, T. (2012). Full face poetry generation. InProceedings of the Third International Conference on Computational Creativity(pp. 95-102).