Computational Creativity Tools for Songwriters

Burr Settles
Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA 15213 USA
bsettles@cs.cmu.edu

Abstract

This paper describes two natural language processing systems designed to assist songwriters in obtaining and developing ideas for their craft. *Titular* is a text synthesis algorithm for automatically generating novel song titles, which lyricists can use to back-form concepts and narrative story arcs. *LyriCloud* is a word-level language "browser" or "explorer," which allows users to interactively select words and receive lyrical suggestions in return. Two criteria for creativity tools are also presented along with examples of how they guided the development of these systems, which were used by musicians during an international songwriting contest.

1 Introduction

Writing lyrics for popular music is challenging. Even apart from musical considerations, well-crafted lyrics should succinctly tell a story, express emotion, or create an image in the listener's mind with vivid and original language. Finding the right words, from an evocative title to a refrain, hook, or narrative detail, is often difficult even for full-time professional songwriters.

This paper considers the task of creating "intelligent" interactive lyric-writing tools for musicians. As a preliminary case study, I discuss two systems that (1) suggest novel song titles and (2) find interesting words given a seed word as input, and deployed them online for participants in an international songwriting event called FAWM[1] (February Album Writing Month). The goal of this event—which attracts both professional and amateur musicians worldwide—is to compose 14 new works of music (roughly an album's worth) during the month of February. Working at a rate of one new song every two days on average is taxing, described by some participants as a "musical marathon." To maintain pace, songwriters are constantly looking for new ideas, which makes them good candidates for using computational creativity tools. Typical lyric-writing tools consist of rhyme or phrase dictionaries which are searchable but otherwise static, passive resources. By contrast, we wish to develop advanced software which uses learned linguistic knowledge to actively help stimulate creative thought.

To formalize the task of developing computational creativity tools, let us first define *creativity* as "the ability to extrapolate beyond existing ideas, rules, patterns, interpretations, etc., and to generate meaningful new ones." By this working definition, which is similar to Zhu et al. (2009), tools that assist humans in creative endeavors should:

1. Suggest instances *unlike* the majority that exist. If one were to model instances statistically, system proposals should be "outliers."

2. Suggest instances that are *meaningful*. Purely random proposals might be outliers, but they are not likely to be interesting or useful.

Previous approaches to linguistic lyric-modeling have generally not focused on creativity, but rather on quantifying "hit potential" (Yang et al., 2007), which is arguably the opposite, or classifying musical genre (Li and Ogihara, 2004; Neumayer and

[1] http://fawm.org

Proceedings of the NAACL HLT 2010 Second Workshop on Computational Approaches to Linguistic Creativity, pages 49–57,
Los Angeles, California, June 2010.

Rauber, 2007). There has been some work on automatically generating percussive lyrics to accompany a given piece of musical input (Oliveira et al., 2007; Ramakrishnan et al., 2009), and there exists a rich body of related work on natural language generation for fiction (Montfort, 2006; Solis et al., 2009), poetry (Gervás, 2001; Manurung, 2004; Netzer et al., 2009), and even jokes (Binstead, 1996). However, the goal of these systems is to be an "artificial artist" which can create complete works of language autonomously, rather than interactive tools for assisting humans in their creative process.

A few computational lyric-writing tools have been developed outside of academia, such as *Verbasizer*, which was famously co-created by rock star David Bowie to help him brainstorm ideas (Thompson, 2007). These types of systems take a small amount of seed text as input, such as a newspaper article, and generate novel phrases by iterating through random word permutations. However, these approaches fail the second criterion for creativity tools, since the majority of output is not meaningful. Many other so-called lyric generators exist on the Internet[2], but these are by and large "Mad-Lib" style fill-in-the-blanks meant for amusement rather than more serious artistic exploration.

The contribution of this work is to develop and study methods that satisfy both criteria for computational creativity tools—that their suggestions are both *unlikely* and *meaningful*—and to demonstrate that these methods are useful to humans in practice. To do this, I employ statistical natural language models induced from a large corpus of song lyrics to produce real-time interactive programs for exploring songwriting ideas. The rest of this paper is organized as follows. Section 2 describes the overall approach and implementation details for these tools. Section 3 presents some empirical and anecdotal system evaluations. Section 4 summarizes my findings, discusses limitations of the current tools, and proposes future directions for work in this vein.

2 Methodology

Davis (1992) describes the discipline of songwriting as a multi-step process, beginning with *activation* and *association*. Activation, according to Davis, involves becoming hyper-observant, embracing spontaneous ideas, and then choosing a title, theme, or progression around which to build a song. Association is the act of expanding on that initial theme or idea as much as possible. Subsequent steps are to develop, organize, and winnow down the ideas generated during the first two stages.

Following this philosophy, I decided to design two natural language processing systems aimed at stimulating songwriters during the early stages of the process, while adhering to the criteria for creativity tools defined in Section 1. I call these tools *Titular*, which suggests novel song titles as a starting point, and *LyriCloud*, which expands lyrical ideas by suggesting related words in the form of a "cloud."

To encourage songwriters to actually adopt these tools, they were deployed online as part of the official FAWM website for participants to use. This posed some development constraints, namely that the user interface be implemented in the PHP language and run in a shared web-hosting environment. Because of this setup, complex inference methods based on a large number of statistics had to be avoided in order to maintain speed and interactivity. Thus, I was limited to approaches where sufficient statistics could be pre-computed and stored in a database to be accessed quickly as needed.

To induce linguistic models for Titular and LyriCloud, existing song data were needed for training. For this, I used a "screen scraper" to extract song title, artist, and lyric fields from ubiquitous online lyrics websites. In this way, I collected a corpus of 137,787 songs by 15,940 unique artists. The collection spans multiple genres (e.g., pop/rock, hip-hop, R&B, punk, folk, blues, gospel, showtunes), and has good coverage of mainstream charting artists (e.g., Beyoncé, R.E.M., Van Halen) as well as more obscure independent performers (e.g., Over the Rhine, Dismemberment Plan). An estimated 85% of the songs are primarily in English.

2.1 Titular

Figure 1 shows example output from Titular, which presents the user with five suggested song titles at a time. Suggestions often combine visceral and contradictory images, like "Bleeding in the Laughter," while others lend themselves to more metaphorical interpretation, such as "Heads of Trick" (which

[2] http://www.song-lyrics-generator.org.uk

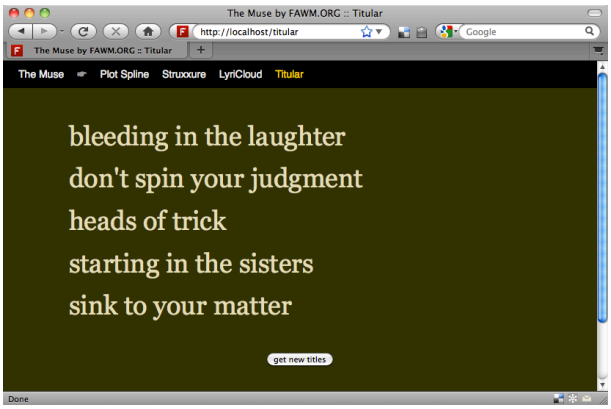

Figure 1: A screenshot of the Titular user interface.

Learned Template	Freq.	Example Source Title
JJ NN	3531	Irish Rover
VB	783	Breathe
VBN NN	351	Born Yesterday
NN and NN	205	Gin And Juice
NNP POS NN	167	Tom's Diner
FW NN	65	Voodoo Woman
VB and VB	57	Seek And Destroy
CD JJ NNS	49	99 Red Balloons
JJ , JJ NN	14	Bad, Bad Girl
you 're so JJ	8	You're So Vain
NN (the NN)	7	Silver (The Hunger)
VBG with a NN	4	Walking With A Zombie

Table 1: Example title templates induced by Titular.

evokes some sort of executive committee for deceit). Occasionally the output is syntactically valid, but a little too awkward to interpret sensibly, such as "Starting in the Sisters."

To accomplish this, I adopted a *template-based* approach (Deemter et al., 2005) to title synthesis. Instead of using hand-crafted templates, however, Titular learns its own using the following method:

1. Source titles in the training corpus are tokenized and tagged for part-of-speech (POS). Input is first lowercased to discourage the tagger from labeling everything NNP (proper noun).

2. The *open* word classes (i.e., various forms of adjectives, adverbs, nouns, and verbs) are substituted with their assigned POS tag, while *closed* words classes (i.e., conjunctions, determiners, prepositions, pronouns and punctuation) remain intact. Titular also remembers which words were substituted with which tag, in order to use them in generating new titles in the future. Vulgar and offensive words are filtered out using regular expressions.

3. The induced templates and words are culled if they occur in the corpus below a certain frequency. This step helps to counterbalance tagging errors, which is important because song titles are often too short to infer POS information accurately. Thresholds are set to 3 for templates and 2 for POS-word pairs.

Table 1 shows several example templates induced using this method. To generate new titles, Titular first randomly selects a template in proportion to its empirical frequency, and likewise selects words to replace each POS tag in the template (drawn from the set of words for the corresponding tag). This model can be thought of as a simple stochastic context-free grammar (Lari and Young, 1990) with only a small set of production rules. Specifically, the root nonterminal S goes to complete templates, e.g.,

$$S \rightarrow \text{VBG , VBG} \mid \text{UH , JJ NN !} \mid \ldots,$$

and POS nonterminals go to words that have been tagged accordingly in the corpus, e.g.,

$$\begin{aligned} \text{VBG} &\rightarrow \text{waiting} \mid \text{catching} \mid \text{going} \mid \ldots, \\ \text{JJ} &\rightarrow \text{good} \mid \text{little} \mid \text{sacrificial} \mid \ldots, \\ \text{NN} &\rightarrow \text{time} \mid \text{life} \mid \text{dream} \mid \ldots, \\ \text{UH} &\rightarrow \text{hey} \mid \text{oh} \mid \text{yeah} \mid \ldots, \end{aligned}$$

all with corresponding probabilities based on frequency. Using these production rules, Titular can generate novel titles like "Going, Waiting" or "Oh, Little Life!" The system learned 2,907 template production rules and 11,247 word production rules from the lyrics corpus, which were stored with frequency information in a MySQL database for deployment. Post-processing heuristics are implemented in the user interface to strip white-spaces from punctuation and contractions to improve readability. Output remains lowercased as an aesthetic decision.

An alternative approach to title generation is an n-gram model based on *Markov chains*, which are common for both natural language generation (Jurafsky and Martin, 2008) and music synthesis (Farbood and Schoner, 2001). For titles, each word w_i is generated with conditional probability

$P(w_i|w_{i-n}, \ldots, w_{i-1})$ based on the n words generated previously, using statistics gathered from titles in the lyrics corpus. However, this approach tends to simply recreate titles in the training data. In a preliminary study using $n = \{1, 2, 3, 4\}$ and 200 titles each, the proportion of suggestions that were verbatim recreations of existing titles ranged from 35% to 97%: ▂▅▉. When n-gram titles do manage to be novel, they tend to be long and unintelligible, such as "Too Many A Woman First the Dark Clouds Will It Up" or "Born of Care 4 My Paradise." These two extremes satisfy one or the other, but not both of the criteria for creativity tools. By contrast, none of the 200 template-generated titles existed in the source database, and they tend to be more intelligible as well (see results in Section 3.1).

The template grammar offers several other advantages over the n-gram approach, including fewer inference steps (which results in fewer database queries) and helping to ensure that titles are well-formed with respect to longer-range syntactic dependencies (like matching parentheses). Constraining words by part-of-speech rather than immediate context also allows the system to create novel and unexpected word juxtapositions, which satisfies the first criterion for creativity tools, while remaining relatively coherent and meaningful, which helps satisfy the second criterion.

2.2 LyriCloud

Figure 2 shows example output from LyriCloud, which takes a *seed* (in this case, the word "dream" highlighted near the center) and suggests up to 25 related words arranged visually in a *cloud* (Bateman et al., 2008). The size of each word is intended to communicate its specificity to the cloud.

Notice that LyriCloud's notion of related words can be quite loose. Some suggestions are modifiers of the seed, like "broken dream" and "deep dream," although these invoke different senses of dream (metaphorical vs. literal). Some suggestions are part of an idiom involving the seed, such as "fulfill a dream," while others are synonyms/antonyms, or specializations/generalizations, such as "nightmare." Still other words might be useable as rhymes for the seed, like "smithereen" and even "thing" (loosely). The goal is to help the user see the seed word in a variety of senses and perspectives. Users

Figure 2: A screenshot of the LyriCloud user interface.

may also interact with the system by clicking with a pointer device on words in the cloud that they find interesting, and be presented with a new cloud based on the new seed. In this way, LyriCloud is a sort of "language browser" for lyrical ideas.

I liken the problem of generating interesting word clouds to an information retrieval task: given a seed word s, return a set of words that are related to s, with the constraint that they not be overly general terms. To do this, the corpus was pre-processed by filtering out common stop-words and words that occur in only one song, or fewer than five times in the entire corpus (this catches most typos and misspellings). As with Titular, vulgar and offensive words are filtered using regular expressions.

For a potential seed word s, we can compute a similarity score with every other word w in the corpus vocabulary using the following measure:

$$\text{sim}(s, w) = \Big(1 + \log c(s, w)\Big) \cdot \log \frac{N}{u(w)},$$

where $c(s, w)$ is the number of times s and w co-occur in the same *line* of a lyric in the corpus, $u(w)$ is the number of unique words with which w occurs in any line of the corpus, and N is the size of the overall vocabulary. This is essentially the well-known log-tempered *tf·idf* measure from the information retrieval literature, if we treat each seed s as a "document" by concatenating all the lines of text in which s appears.

I also experimented with the co-occurence frequency $c(s, w)$ and *point-wise mutual information* (Church and Hanks, 1989) as similarity functions.

The former still used overly common words (e.g., "love," "heart," "baby"), which fails the first criterion for creativity tools, and the latter yielded overly seed-specific results (and often typos not filtered by the pre-processing step), which can fail the second criterion. The log-tempered *tf·idf* metric provided a reasonable balance between the two extremes.

To generate a new cloud from a given seed, up to 25 words are randomly sampled from the top 300 ranked by similarity, which are pre-computed and stored in the database for efficient lookup. The sampled words are then sorted alphabetically for display and scaled using a polynomial equation:

$$\text{size}(w) = f_0 + f_1 \cdot \left(1 - \frac{\text{rank}(w)}{K}\right)^4,$$

where f_0 and f_1 are constants that bound the minimum and maximum font size, and K is the number of words in the cloud (usually 25, unless there were unusually few words co-occuring with s). This choice of scaling equation is ad-hoc, but produces aesthetically pleasing results.

As a point of comparison, the original approach for LyriCloud was to employ a *topic model* such as Latent Dirichlet Allocation (Blei et al., 2003). This is an unsupervised machine learning algorithm that attempts to automatically organize words according to empirical regularities in how they are used in data. Table 2 shows 15 example "topics" induced on the lyrics corpus. Intuitively, these models treat documents (song lyrics) as a probabilistic mixture of topics, which in turn are probabilistic mixtures of words. For example, a religious hymn that employs nature imagery might be a mixture of topics #11 and #13 with high probability, and low probability for other topics (see Blei et al. for more details).

To generate clouds, I trained a model of 200 topics on the lyrics corpus, and let the system choose the most probable topic for a user-provided seed. It then randomly sampled 25 of the 300 most probable words for the corresponding topic, which were scaled in proportion to topic probability. The intuition was that clouds based on the top-ranked words for a topic should be semantically coherent, but sampling at random from a large window would also allow for more interesting and unusual words.

In a small pilot launch of the system, songwriters rejected the topic-based version of LyriCloud for two reasons. First, the top-ranked words within a topic tend to be high frequency words in general (consider Table 2). These were deemed by users to be unoriginal and in violation the first criterion for creativity tools. Tweaking various system parameters, such as the number of topics in the model, did not seem to improve performance in this respect. Second, while words were thought to be coherent overall, users had trouble seeing the connection between the chosen topic and the highlighted seeds themselves (see also the results in Section 3.1). Instead, users wanted to receive interesting suggestions that were more specifically tailored to the seeds they chose, which motivated the information-retrieval view of the problem. While this is plausible using topic models with a more sophisticated sampling scheme, it was beyond the capabilities of a simple PHP/MySQL deployment setup.

#	Most Probable Words By Topic
1	love baby give true sweet song girl ...
2	la big black white hair wear hot ...
3	hold hand hands head free put back ...
4	love find nothing something everything wrong give ...
5	yeah baby hey man girl rock ooh ...
6	feel make eyes gonna cry makes good ...
7	fall turn light face sun again world ...
8	night day cold long sleep dream days ...
9	mind world lose nothing left gonna shake ...
10	time life long live day end die ...
11	god lord man hell king heaven jesus ...
12	hear play call people good talk heard ...
13	sky eyes fire fly blue high sea ...
14	heart inside pain soul break broken deep ...
15	go back home again time gonna coming ...

Table 2: Example words from a topic model induced by Latent Dirichlet Allocation on the lyrics corpus.

3 Results

This section discusses some results and observations about Titular and LyriCloud.

3.1 Empirical Evaluation

I conducted an empirical study of these systems using Amazon Mechanical Turk[3], which is being used increasingly to evaluate several systems on open-ended tasks for which gold-standard evaluation data does not exist (Mintz et al., 2009; Carlson et al.,

[3] http://www.mturk.com

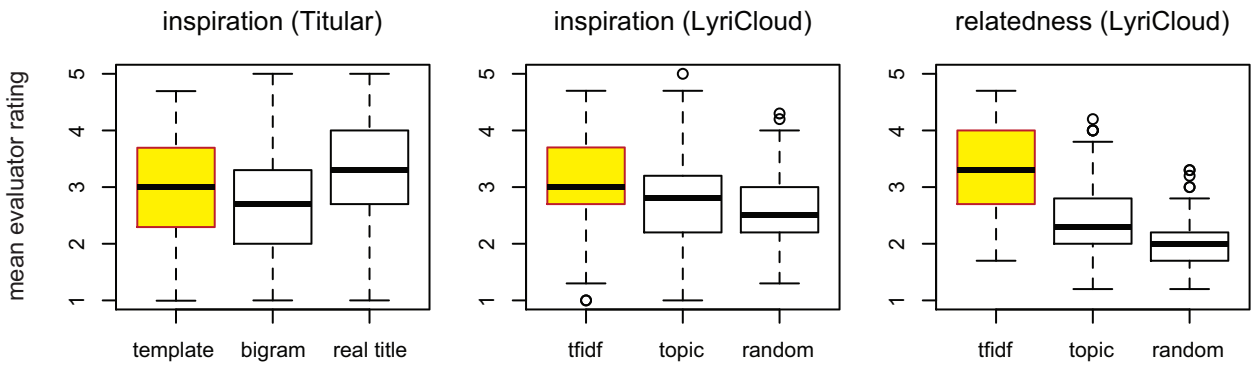

Figure 3: Box plots illustrating the distribution of "inspiration" and "relatedness" ratings assigned by Mechanical Turk evaluators to Titular and LyriCloud output. Boxes represent the middle 50% of ratings, with the medians indicated by thick black lines. Whiskers on either side span the first and last quartiles of each distribution; circles indicate possible outliers. The methods actually deployed for use online are are highlighted on the left of each plot.

Method	Good Title?	Grammatical?	Offensive?
template	68%	67%	6%
bigram	62%	52%	7%
real title	93%	88%	9%

Table 3: Titular results from Mechanical Turk evaluators.

2010). This was done because (1) we would like some quantification of how well these systems perform, and (2) using Mechanical Turk workers should provide more stringent and objective feedback than the songwriters for whom the tools were developed.

For each tool, I generated 200 instances (i.e., titles for Titular and word clouds for LyriCloud) and had evaluators fill out a simple form for each, composed of yes/no questions and ratings on a five-star scale. Each instance was given to three unique evaluators in order to break ties in the event of disagreement, and to smooth out variance in the ratings. Titular is compared against titles generated by a bigram Markov chain model ($n = 1$), as well as actual song titles randomly selected from the lyrics database. LyriCloud is compared against the topic model method, and a baseline where both words and their sizes are generated at random.

Table 3 summarizes Titular results for the yes/no questions in the Mechanical Turk evaluation. At least two of three evaluators agreed that 68% of the suggested template-based phrases would make good song titles, which is higher than the bigram method, but lower than the 93% rate for real song titles (as expected). Similarly, template-based suggestions were deemed more grammatical than the bigram approach and less grammatical than actual songs. Somewhat surprisingly, 12% of all titles in the evaluation were judged to be good titles despite grammatical errors, including "Dandelion Cheatin" and "Dressed in Fast." This is an interesting observation, indicating that people sometimes enjoy titles which they find syntactically awkward (perhaps even *because* they are awkward). All methods yielded a few potentially offensive titles, which were mostly due to violent or sexual interpretations.

Evaluators were also asked to rate titles on a scale from *boring* (1) to *inspiring* (5), the results of which are shown on the left of Figure 3. The template approach does fairly well in this respect: half its suggestions are rated 3 or higher, which is better than the bigram method but slightly worse than real titles. Interestingly, distributional differences between the ratings for template-based output and actual song titles are not statistically significant, while all other differences are significant[4]. This suggests that Titular's titles are nearly as good as real song titles, from an inspirational perspective. Unlike other methods, no single template-based title received a unanimous rating of 5 from all three evaluators (although "The Lungs Are Pumping" and "Sure Spider Vengeance" both received average scores of 4.7).

For LyriCloud, evaluators were asked to rate both the level of inspiration and whether or not they found the words in a cloud to be *unrelated* (1) or

[4]Kolmogorov-Smirnov tests with $p < 0.05$. Multiple tests in this paper are corrected for using the Bonferroni method.

related (5). These results are shown in the center and right plots of Figure 3. In both measures, the *tf·idf* approach outperforms the topic model, which in turn outperforms a random collection of words. All differences are statistically significant. Interestingly, the R^2 coefficient between these two measures is only 0.25 for all word clouds in the evaluation, meaning that relatedness ratings only explain 25% of the variance in inspiration ratings (and vice versa). This curious result implies that there are other, more subtle factors at play in how "inspiring" humans find a set of words like this to be. Additionally, only 36% of evaluators reported that they could find meaning in the size/scale of words in *tf·idf* clouds (compared to 9% for the topic model and 1% for random), which is lower than anticipated.

3.2 Anecdotal Results

The implementations described in this paper were developed over a four-day period, and made available to FAWM participants mid-way through the songwriting challenge on February 16, 2010. They were promoted as part of a suite of online tools called *The Muse*[5], which also includes two other programs, *Struxxure* and *Plot Spline*, aimed at helping songwriters consider various novel song structures and plot constraints. These other tools do not use any language modeling, and a discussion of them is beyond the scope of this paper.

Table 4 summarizes the internet traffic that The Muse received between its launch and the end of the FAWM challenge on March 1, 2010 (thirteen days). Encouragingly, Titular and LyriCloud were the most popular destinations on the website. These statistics include visitors from 36 countries, mostly from the United States (55%), Germany (15%) and the United Kingdom (9%). News of these tools spread well beyond the original FAWM community, as 29% of website visitors were referred via hyperlinks from unaffiliated music forums, songwriting blogs, and social websites like Facebook and Twitter.

FAWM participants typically post their songs online after completion to track their progress, so they were asked to "tag" the songs they wrote with help from these creativity tools as a way to measure how much the tools were being used in practice. By the end of the challenge, 66 songs were tagged "titular" and 29 songs were tagged "lyricloud." Note that these figures are conservative lower-bounds for actual usage, since not all participants tag their songs and, as previously stated, a significant portion of the internet traffic for these tools came from outside the FAWM community.

Tool	Pageviews	% Traffic
Titular	11,408	42.9%
LyriCloud	5,313	20.0%
Struxxure	4,371	16.5%
Plot Spline	3,219	12.1%
Home Page	2,248	8.5%
Total	26,559	100.0%

Table 4: Website activity for The Muse tools between February 16 and March 1, 2010. The creativity tools discussed in this paper are italicized.

The tools were published without detailed instructions, so songwriters were free to interpret them however they saw fit. Several users found inspiration in Titular titles that are interpretable as metaphor or synecdoche. For example, Expendable Friend (the stage name of British singer/songwriter Jacqui Carnall) wrote a song around the suggestion "I Am Your Adult," which she interpreted this way:

> So, this is a song about the little voice inside you that stops you from doing fun things because you're not a child any more and you can't really get away with doing them.

Surprisingly, even non-lyricists adopted Titular. "Mexican of No Breakup" led guitarist Ryan Day to compose a latin-style instrumental, and "What a Sunset!" became an ambient electronic piece by an artist by the pseudonym of Vom Vorton.

While LyriCloud was about half as popular as Titular, it was arguably open to a wider range of interpretation. Some songwriters used it as originally envisioned, i.e., a "browser" for interactively exploring lyrical possibilities. New York lawyer and folksinger Mike Skliar wrote two songs this way. He said this about the process:

> I had a few images in mind, and I would type in the key word of the image, which generated other words sometimes slightly related... then kind of let my mind free associate on what those words might mean juxtaposed against

[5] http://muse.fawm.org

> the first image, and came up with about half the song that way.

Unexpectedly, some took LyriCloud as a challenge to write a song using *all* the words from a single cloud (or as many as possible), since it chooses a seed word at random if none is provided as input. Songwriter James Currey, who actually used LyriCloud and Titular together to write "For the Bethlehem of Manhattan," described the process this way:

> It was like doing a puzzle, and the result is actually quite surprisingly coherent AND good.

As a final anecdote, middle school English teacher Keith Schumacher of California was a FAWM 2010 participant. He shared these tools with faculty members at his school, who designed an in-class creative writing exercise for students involving words generated by LyriCloud, projected overhead in the classroom. This demonstrates the utility of these and similar tools to a broad range of age groups and writing styles.

4 Conclusions and Future Work

In this paper, I introduced the task of designing computational creativity tools for songwriters. I described two such tools, *Titular* and *LyriCloud*, and presented an empirical evaluation as well as anecdotal results based on actual usage during an international songwriting event. I also presented two criteria for creativity tools—that their suggestions be both *unlikely* and *meaningful* to the human artists interacting with them—and showed how these principles guided the development of the tools.

This preliminary foray into creativity tools based on language modeling shows the potential for creativity-oriented human-computer interaction. However, there is still much that can be improved on. For example, Titular might benefit from training with a more traditional context-free grammar (i.e., one with recursive production rules), which might yield more complex and interesting possibilities. LyriCloud could be extended to include bigram and trigram phrases in addition to single words. Also, the vocabularies of both systems might currently suffer due to the limited and domain-specific training data (the lyrics corpus), which could be supplemented with other sources.

Perhaps more importantly, neither of the current tools incorporate any explicit notion of wordplay (e.g., rhyme, alliteration, assonance, puns) or any other device of creative writing (meter, repetition, irony, etc.). The systems occasionally do suggest instances that embody these properties, but they are wholly by chance at this stage. One can, however, imagine future versions that take preference parameters as input from the user, and try to suggest instances based on these constraints.

Acknowledgments

Thanks to Jacob Eisenstein for helpful discussions during the development of these tools, and to the anonymous reviewers for valuable suggestions that much improved the paper. To all the "fawmers" who inspired the project, used the tools, and provided feedback—particularly those who gave permission to be discussed in Section 3.2—you rock.

References

S. Bateman, C. Gutwin, and M. Nacenta. 2008. Seeing things in the clouds: the effect of visual features on tag cloud selections. In *Proceedings of the ACM Conference on Hypertext and Hypermedia*, pages 193–202. ACM.

K. Binstead. 1996. *Machine humour: An implemented model of puns*. Ph.D. thesis, University of Edinburgh.

D.M. Blei, A.Y. Ng, and M. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

A. Carlson, J. Betteridge, R. Wang, E.R. Hruschka Jr, and T. Mitchell. 2010. Coupled semi-supervised learning for information extraction. In *Proceedings of the International Conference on Web Search and Data Mining (WSDM)*, pages 101–110. ACM Press.

K. Church and P. Hanks. 1989. Word associate norms, mutual information and lexicography. In *Proceedings of the Association for Computational Linguistics (ACL)*, pages 76–83. ACL Press.

S. Davis. 1992. *The Songwriters Idea Book*. Writer's Digest Books.

K. Deemter, M. Theune, and E. Krahmer. 2005. Real versus template-based natural language generation: a false opposition? *Computational Linguistics*, 31(1):15–24.

M. Farbood and B. Schoner. 2001. Analysis and synthesis of Palestrina-style counterpoint using Markov chains. In *Proceedings of the International Computer*

Music Conference, pages 471–474. International Computer Music Association.

P. Gervás. 2001. An expert system for the composition of formal spanish poetry. *Journal of Knowledge-Based Systems*, 14:181–188.

D. Jurafsky and J.H. Martin. 2008. *Speech and Language Processing*. Prentice Hall.

K. Lari and S. J. Young. 1990. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35–56.

T. Li and M. Ogihara. 2004. Music artist style identification by semi-supervised learning from both lyrics and content. In *Proceedings of the ACM International Conference on Multimedia*, pages 364–367. ACM.

H. Manurung. 2004. *An evolutionary algorithm approach to poetry generation*. Ph.D. thesis, University of Edinburgh.

M. Mintz, S. Bills, R. Snow, and D. Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Association for Computational Linguistics (ACL)*, pages 1003–1011. ACL Press.

N. Montfort. 2006. Natural language generation and narrative variation in interactive fiction. In *Proceedings of the AAAI Workshop on Computational Aesthetics*.

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

R. Neumayer and A. Rauber. 2007. Integration of text and audio features for genre classification in music information retrieval. In *Proceedings of the European Conference on Information Retrieval (ECIR)*, pages 724–727. Springer.

H.R.G. Oliveira, F.A. Cardoso, and F.C. Pereira. 2007. Tra-la-lyrics: An approach to generate text based on rhythm. In *Proceedings of the International Joint Workshop on Computational Creativity, London*.

A. Ramakrishnan, S. Kuppan, and S. Lalitha Devi. 2009. Automatic generation of tamil lyrics for melodies. In *Proceedings of the Workshop on Computational Approaches to Linguistic Creativity (CALC)*, pages 40–46. ACL Press.

C. Solis, J.T. Siy, E. Tabirao, and E. Ong. 2009. Planning author and character goals for story generation. In *Proceedings of the Workshop on Computational Approaches to Linguistic Creativity (CALC)*, pages 63–70. ACL Press.

D. Thompson. 2007. *Hallo Spaceboy: The Rebirth of David Bowie*. ECW Press.

J.M. Yang, C.Y. Lai, and Y.H. Hsieh. 2007. A quantitative measurement mechanism for lyrics evaluation: A text mining approach. In *Proceedings of the International Conference on Business and Information (BAI)*. ATISR.

X. Zhu, Z. Xu, and T. Khot. 2009. How creative is your writing? In *Proceedings of the Workshop on Computational Approaches to Linguistic Creativity (CALC)*, pages 87–93. ACL Press.