Spring 2018

AUTOMATED LYRICAL NARRATIVE WRITING

Divya Singh
San Jose State University

Recommended Citation
Singh, Divya, "AUTOMATED LYRICAL NARRATIVE WRITING" (2018). *Master's Projects*. 617.
DOI: https://doi.org/10.31979/etd.756z-tvbf
https://scholarworks.sjsu.edu/etd_projects/617

AUTOMATED LYRICAL NARRATIVE WRITING

A Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By

Divya Singh

May 2018

The Designated Project Committee Approves the Project Titled

AUTOMATED LYRICAL NARRATIVE WRITING

by

Divya Singh

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSÉ STATE UNIVERSITY

May 2018

Dr. Sami Khuri	Department of Computer Science
Dr. Philip Heller	Department of Computer Science
Dr. Margareta Ackerman	Department of Computer Engineering, Santa Clara University

ABSTRACT

AUTOMATED LYRICAL NARRATIVE WRITING

by Divya Singh

Computational Creativity studies the potential of computers to act as autonomous creators and co-creators in addition to tools helping people. Creativity is evident in music, visual art, problem solving and languages. Significant work has been conducted in the area of linguistic creation mainly in the generation of stories, puns, rhymes, jokes, similes, and poetry. One of the major challenges of computational creativity is to generate lyrics that exhibit human-level creativity. On one hand, the lyrics generated should be meaningful and coherent, while on the other hand, they should satisfy poetry constraints such as rhyme scheme, rhyme type, and the number of syllables. The goal of this project is to combine the two art forms of storytelling and lyrics writing through the automated creation of coherent lyrics. The project also highlights the approaches to the creation of poetry and lyrics. The resulting model is named MexicA's BaLlad MachinE (MABLE). This is the first computational system that generates narrative-based lyrics.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to Dr. Sami Khuri whose motivation kept me engaged in my research.

I would also like to thank Dr. Philip Heller for his valuable suggestions and for serving on my defense committee.

I would like to thank Dr. Margareta Ackerman for her guidance and motivation during development of this research work. Under her supervision, the work in this project resulted in a publication titled “A Ballad of the Mexicas: Automated Lyrical Narrative Writing,” presented at the International Conference on Computational Creativity, 2017. Without her persistent help, this paper would not have been possible. Also, I wish to acknowledge the valuable advice given by Rafael Perez y Perez during the process.

I genuinely thank my parents for their love, support, and prayers that kept me moving forward.

Most importantly, I wish to thank my brother for supporting me and believing in me.

TABLE OF CONTENTS

LIST OF FIGURES

I. INTRODUCTION

No single definition can do justice to poetry as it varies across genres and one can always find an example that contradicts any given definition. However, Levin (1962) defines poetry as a literary form in which language is used in a concentrated blend of sound and imagery to create an emotional response [1]. This definition points to the main ingredients of a poem which are content and emotions.

Creating a meaningful story not only requires correct grammatical structure but also an understanding of theme, plot, and characters. Unsurprisingly, automated creation of coherent stories persists as a significant challenge. This can be seen in previous work on lyrics and poetry generation. Despite these lyrics being deficient in coherency, they were emotionally engaging due to the selection of related and powerful words. On the other hand, lyrics written by humans across diverse genres often center around a consistent story.

The main focus of this project is to integrate automated storytelling with lyrics writing. The goal of this integration is to tackle the challenges faced in both of these domains and create emotionally engaging lyrics that narrate a story to the audience.

The contributions of this work are as follows:

1. This new approach of automating the writing of ballads has not been previously utilized by either machines or humans. The method

starts with a complete plot line generated by MEXICA, capturing the main elements of the story. The next step is then used to endow the story with rhyme and rhythm. Specifically, every other line in lyrics is devoted to the progression of the narrative, while the remaining phrases aim to transform the narrative into a ballad.

2. In the first phase, MABLE (MexicA's BaLlad MachinE) relies on MEXICA, a plot generation machine based on the engagement-reflection model for creative writing.
3. The second phase utilizes a statistical model to expand the plot into lyrics. A second-order Markov model, trained on a corpus of love songs, is used to generate candidate sentences that rhyme with the original sentence and following its metric structure which leads to the seamless integration of the narrative into the ballad.

The remainder of this report is organized as follows. Section II describes the background. Section III covers a summary of related previous work. Section IV gives a brief introduction to MEXICA, which is integral to MABLE. Section V gives an in-depth description of MABLE and a step-by-step example for clear understanding of workflow of MABLE. Finally, Section VI discusses the conclusion and future work.

II. BACKGROUND

A. Markov Chain

A Markov chain is a stochastic process that consists of a sequence of states where the probability distribution of a state at time t+1 depends on the state at time t and not on the preceding states, and t = 0, 1, 2, 3 ... [2]. Each state has a transition probability of going from one state to another state.

In mathematical terms, Markov process can be represented in the form of a triplet (Q, p, A), where Q is a finite set of states, p is the initial state probabilities and A is the state transition probabilities. Let q_i be the probability that the system is in state i at time 0, P be the transition probability matrix, and $\sum_{j=1 \text{ to } s} p_{ij} = 1$ [2, eq(3)].

Then, the probability of a sequence of states $X_1, X_2, \ldots, X_T$ at time 1, …, T is given as follows:

$$P(X_1, X_2, \ldots, X_T) = q_{x1} \pi^{T}{}_{t=2} P_{xt-1xt} \qquad [2, eq(4)]$$

The probability of each symbol X_i depends only on the value of the preceding symbol X_{i-1}. A two-state Markov process is shown in Fig. 1. In this figure, the arrow indicates the direction and the number on the arrow shows the probability of going from one state to another. For example, if the Markov process is in state B then the probability that it goes to state A is 0.8 while the probability that it remains in state B is 0.2.

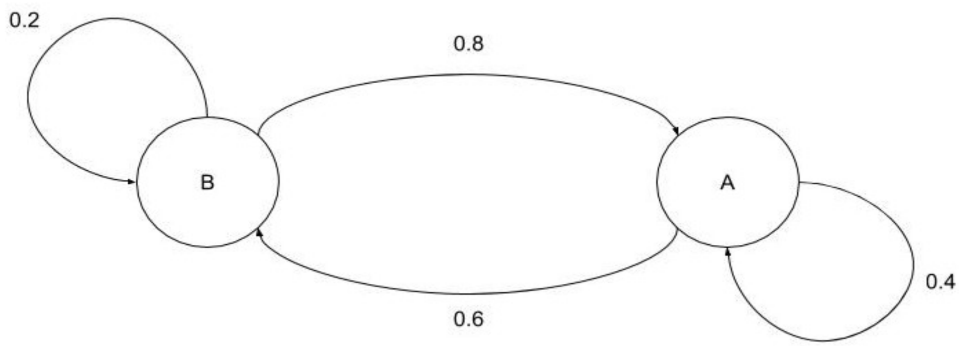

Fig. 1. Two-state Markov chain diagram with the states labeled, B and A.

In a first order Markov chain, the next state depends only on the previous state while in higher order Markov chains, the next state depends on two or more preceding states. Markov chain has been applied in areas such as physics, biology, music, sports, games, etc.

Fig. 2 is the conceptual map of the topics covered in subsequent subsections.

B. Natural Language Processing

Natural Language Processing (NLP) is a field of computer science, artificial intelligence and computational linguistics that is concerned with understanding human language. This field is an active area of research in which computers are taught to understand and manipulate human languages. These languages are used to express knowledge and emotions and convey responses to other people.

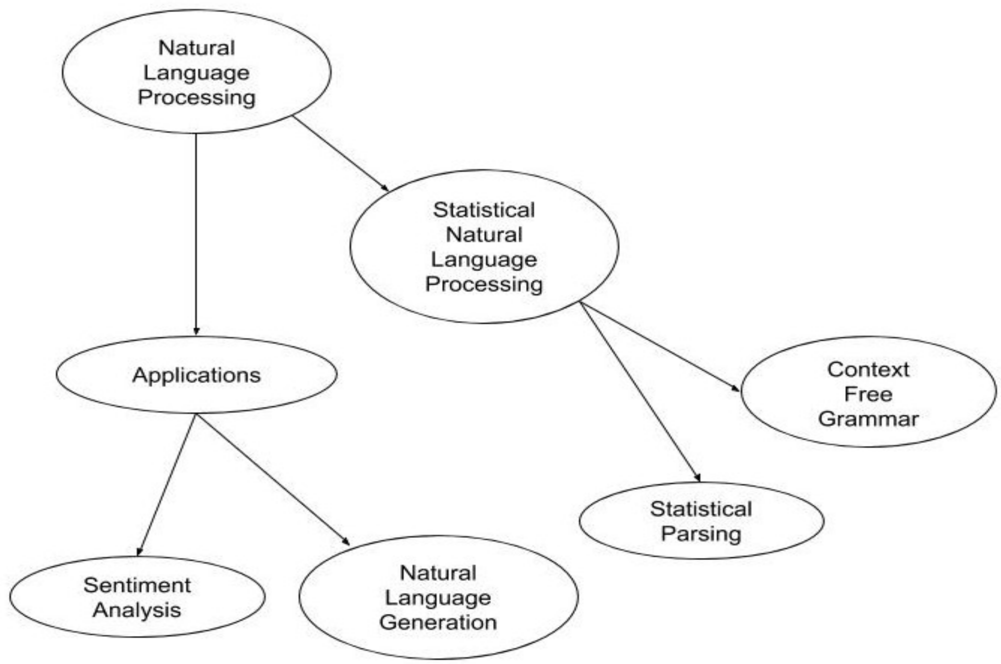

Fig. 2. Subfields of natural language processing covered in this section.

emotions and convey responses to other people. Computers are instructed to behave in a similar manner by processing large language corpora. The applications of NLP include machine translation, natural language text processing, text mining, information retrieval, speech recognition, sentiment analysis, artificial intelligence, and expert systems [3].

1) Statistical Natural Language Processing: Statistical Natural Language Processing is a branch of NLP in which statistical and probabilistic methods are used to resolve problems with the help of data mining and machine learning [4]. They are subfields of artificial intelligence that are trained on huge corpora.

An example of such a model is Markov model where the probability of the next state depends only on the current state. When sentences are long and ambiguous, their processing becomes difficult which requires thousands of possible further analyses [4]. Two such techniques used for natural language generation are described as follows:

a) Stochastic Context-Free Grammar: Context-free grammar (CFG) is a set of grammar rules used to generate languages. These grammar rules are called production rules that consist of terminals, non-terminals, a set of rules, and a start symbol. Stochastic CFG has been used to create novel melodies within a genre [5]. Then, machine learning methods are applied in order to select better results. Poetry parameters such as rhyme, rhythm, and the number of syllables form the basis for constructing lyrics. The process of creating lyrics requires a huge database which can improve the reliability of training the evaluation module [5].

*b) Statistical parsing**:*** Statistical parsing is a process of finding out the most probable parse of a sentence given the probabilities of complete parse for sentences obtained from a corpus of text [6]. Hidden Markov Model and Viterbi Search are types of statistical parsers. They search over an area of all candidates' parses associated with their probabilities and select the most likely parse for the sentence. Candidates are ranked by their probabilities and the most likely interpretation is searched using search and sort algorithms of Artificial Intelligence. The searching is optimized using algorithms like stack search, Viterbi search, and Baum-Welch [6]. For ranking, various machine learning algorithms such as Deep Neural Nets and Recurrent Neural Networks are used.

2) Applications: Applications of NLP are as follows:

a) Sentiment Analysis: Sentiment Analysis, also known as opinion mining, is a classification task to determine the overall attitude (positive, negative or neutral) in the text. It uses NLP to calculate numerical score along with magnitude values. The sentiments can be positive, negative, neutral, good, very good, satisfactory, bad, or very bad. The main objective of sentiment analysis is to improve quality of the product. Numerous approaches determine the sentiments present in word, sentence or document. NLP, machine learning algorithms such as Support Vector Machines, Naive Bayes, or Unsupervised Learning are some of the techniques applied for analyzing sentiments [7]. NLP focuses on using existing natural language processing tools such as PartOfSpeechtagger or Ngrams. P. Goncalves et al. [7] provides comparisons of eight sentiment methods: SentiWordNet, SASA, PANASt, Emoticons, SentiStrength, LIWC, SenticNet, and Happiness Index. No single method gives best results to all the text sources. Therefore, Combined Method is used to blend all the eight sentiment methods and achieve the optimum outcome [7].

b) Natural Language Generation: Natural Language Understanding (NLU) and Natural Language Generation (NLG) are two components of NLP. The former maps the language into representations while the latter extracts meaningful information from those representations. NLG can be viewed as the opposite of NLU [8]. For illustration, Markov text generators create text from a given sample dataset. One of their applications is generating parody. Evaluation of NLG systems involves methods such as test-based evaluation, human ratings, and metrics.

The next section gives an overview of related previous approaches applied to the creation of poetry and lyrics.

III. RELATED WORK

Several attempts have been made at poetry generation ranging from template-based to statistical methods. This section explores such approaches in detail.

Rap Lyric Generator has been designed particularly for the creation of rap lyrics. It is trained on lyrics separated into chorus and verse [9]. These corpora are then used to create two quad gram models that generate sentences which are then combined based on matching syllables in rhyming words. In future work, instead of the quad gram model, Stanford Parser that will create parse trees on different domains of the corpus, different clusters to construct different themes and switch themes from one line to another [9]. Another system that generates rap lyrics is DopeLearning. A corpus of rap lyrics, written by 104 different rap artists, is constructed. It combines lines from the corpus and uses Deep Neural Network and RankSVM to identify the next line in lyrics based on rhyming, structural and semantic similarity [10].

Tra-La-Lyrics 2.0 uses template-based approach for lyrics generation [11]. It combines two previous systems - PoeTryMe and Tra-La-Lyrics 1.0. The former version of Tra-la-lyrics matches stresses in the text with the rhythm of the melody. The new version takes user-provided seed words to create a semantic network that generates candidates. These candidates are then used to fill in a poem template. Tra-la lyrics 2.0 combines these two systems by integrating rhythm information from melodies into PoeTryMe's architecture [11].

Full Face Poetry Generation uses templates provided by the user to construct poems [12]. Based on key phrases extracted from newspaper articles, it selects the mood and combines them with a database of similes to form template-based sentences. A lyricism measure calculates scores on the basis of start rhymes, end rhymes, and syllables count. Next, the poem with the best score is chosen as the poem of the day. In Semi-Automated Lyrics Generation Tool for Mauritian Saga, the user fills the form which is then stored in MySQL database [13]. This information is then extracted to form lyrics. Other work that creates poetry based on user-provided text can be seen in [14] and [15] where the latter one uses stochastic hill-climbing approach.

Scientific Music Generator (SMUG) uses real-world data such as academic papers for creating songs [16]. Rather than extracting words, it extracts features from real songs for creating song structures. The algorithm extracts keywords from the paper and utilizes them for filling one of the song structures in the database. Two Markov chains are implemented where one determines the notes of the melody while the other is used to select the duration of these notes.

Also, there has been significant work done on the automated creation of poetry. Constrained programming is applied for poetry composition [17].

The model consists of two components. The first component, Specifier maintains a static library of constraints that contains information such as rhymes, syllables, number of lines, and number of words on each line. These constraints are then used by the other component, Explorer, to generate poems. Topical poetry is created from a user-specified topic word [18]. This word is then used to identify 1000 similar words which are subsequently grouped in rhyming classes, and rhyming pairs are selected. Next, a final state acceptor is created, where each poem that it could generate satisfies sonnet constraints and the pairs of rhyming words are used to create a rhyming pattern. Finally, the paths are extracted using beam search and Recurrent Neural Networks.

Regardless of substantial work conducted in poetry generation, none of the previous systems aim to create lyrics that convey coherent stories, which is the focus of this project.

IV. MEXICA

MEXICA is a computer model of creativity in writing that generates short stories about old inhabitants of what today is Mexico City [19]. It uses a cycle of engagement and reflection and provides the user the flexibility to explore different aspects of story generation by modifying different parameters.

In MEXICA, a story is represented as a sequence of actions. Each action has extra information stored about it such as preconditions, post conditions in terms of emotional links and tensions between the characters [20]. Tensions reflect conflicts between the actors. They are one of the key elements in short story. Tensions are triggered when a character is murdered, when there is a clash of emotions between characters, or when the health of a character is at risk [20]. Whenever an action is performed in the story, MEXICA looks for any tension between the characters.

Tensions such as Clashing Emotions and Love Competition are called Inferred Postconditions. Clashing Emotion is triggered when a character has two opposite emotions towards another one [20]. For illustration, "A princess falls in love with an enemy." Tension due to Love Competition appears when two actors are in love with a third one.

In this way, MEXICA constructs a story providing additional information in terms of mainly emotional links and tensions that are used by MABLE to extract the sentiments in the story.

The workflow of MABLE is discussed in the next section.

V. IMPLEMENTATION

This section is divided into two parts. The first part describes the architecture of MABLE and the second part presents a step-by-step example creating a figurative sentence for a narrative line.

A. Model Description

MABLE takes the story narrative generated by MEXICA as input. Then, the narrative line is processed by its three main components: Sentence Evaluator, Sentiment Analyzer, and Integrator. In the end, a candidate line is added to each storyline, thus forming lyrics. This process is repeated for each line in the narrative.

The workflow of lyrics generation is divided into three main modules. The first one is Sentence Evaluator which creates a set of sentences that fits with the storylines using poetic elements such as rhyme quality, rhyme scheme, number of syllables, and rhyme type. Then, Sentiment Analyzer filters those phrases using sentiment scores that match with the story's emotions. Finally, Integrator plays its role by substituting the candidate sentences in first person pronouns with third person pronouns. An overview of MABLE's architecture is illustrated in Fig. 3 [21, Fig. 1].

A few linguistic resources are utilized to make lyrics poetic. Phonetic transcription of words is extracted from the CMU Pronunciation Dictionary that contains pronunciations of around 134,000 words. Twinword API

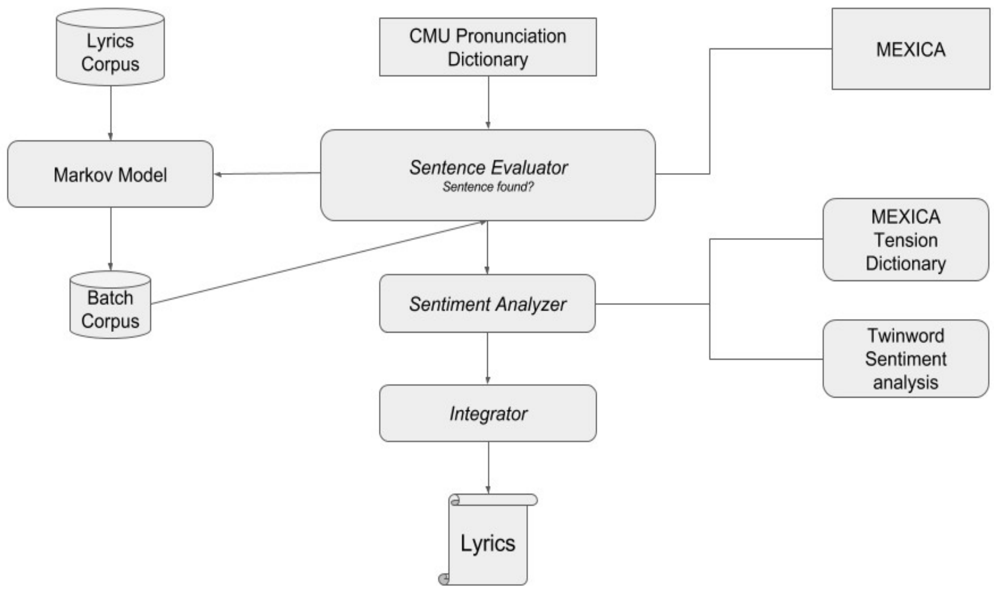

Fig. 3. Overview of lyrics generation workflow.

(https://www.twinword.com/api/) is used to extract the sentiments of phrases which clusters them into positive, negative, and neutral sentiments based on their sentiment scores obtained. All of the components of MABLE are described in detail as follows.

1) Sentence Evaluator: This module exploits the Markov Model in order to generate candidate phrases that rhyme with the narrative line. A pictorial representation of Sentence Evaluator is shown in Fig. 4. A second order Markov model is trained on a corpus of lyrics which are crawled from 70s to 80s rock and pop songs available on azlyrics.com. These 129 songs are sung by various artists and voted by twitter followers. Titles of the top 100 lyrics are taken from an online music catalog (https://www.last.fm/) and the remaining lyrics from an

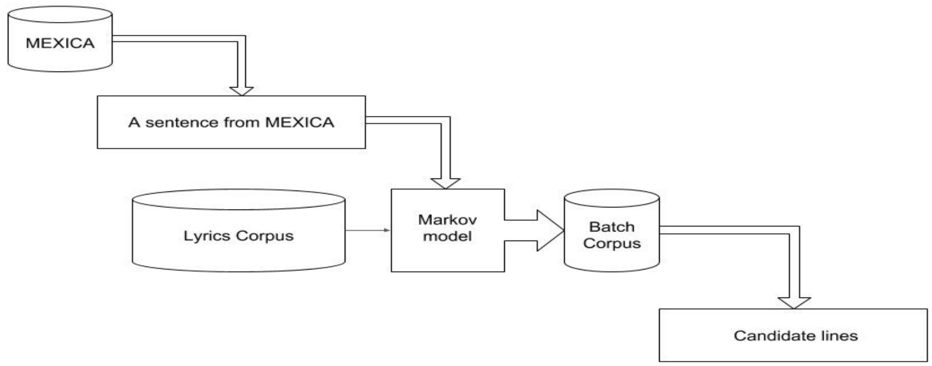

Fig. 4. Flowchart of sentence evaluator.

NME music blog (http://www.nme.com/blogs). After scraping lyrics from the web, they are cleaned by removing punctuations, extra white spaces, and repeated lines.

After getting a narrative line from MEXICA, Sentence Evaluator calls the Markov model repeatedly until 50 high-quality sentences are obtained. The quality of each line is measured on the basis of rhyming score and number of syllables in the line. Every time the Markov model is called, it is configured to generate a batch of around 100 new sentences having no more than 60 characters.

To calculate the rhyming score, a pool of words that rhyme with the last word in the storyline are obtained using NLTK (http://www.nltk.org/).

This pool is then used to search the batch (generated by second-order Markov model) for lines whose last word rhymes with any of the words in the bag. This bag of words is sorted by their rhyme scores. They rhyme with the original to various degrees. Due to this, a lot of candidate lines become acceptable. This idea allows lyrics to more closely resemble human-made songs which are always not so perfect in rhyming.

The process of finding out whether the two words rhyme or not is as follows. The words in a sentence are tokenized using NLTK. Then, Carnegie Mellon University Pronunciation Dictionary is used to get the phonetic transcription of words. The dictionary has 127,069 words and their multiple pronunciations. Out of these words, 119,400 are assigned a unique pronunciation, 6830 have two, and 839 have three or more pronunciations. When any word is not found in the dictionary, the last letter of the word is taken. A tuple list of words containing lexicons and their rhyming scores is formed from CMUdict (http://www.speech.cs.cmu.edu/cgibin/cmudict) for each of the pronunciations. Various degrees of phonetic similarities are considered like slant rhymes, end rhymes, and assonance rhymes. End rhyme occurs when the last syllables of words are matched. It is the most common type of rhyme found in poetry. It is the simplest one therefore makes it easier for the audience to remember it. In slant rhyme, words have similar consonant sounds. Assonance rhymes are formed by words that have matching vowels. MABLE uses an alternating rhyming scheme which is the most frequently used, also known as AABB.

After calculating rhyme scores, the candidate sentences are sorted based on their rhyme quality scores thereby selecting the top 50. These top candidates then go to the next level which involves counting the number of syllables. CMUdict is used to count the number of syllables in words which then can be used to count the total number of syllables in each sentence. In the worst case, if no candidates are obtained, common interjections such as “Oho oh oh oh” matching the number of syllables as that of in the narrative line are returned. These are then passed to the Sentiment Analyzer to get emotionally connected with the story.

2) Sentiment Analyzer: Candidate sentences obtained from the Sentence Evaluator are well-structured but sometimes they get disconnected from the original story so sentiment analysis needs to be incorporated. Fig. 5 shows how sentiments are matched between narrative sentences generated by MEXICA and candidate phrases provided by the Sentence Evaluator.

In this module, a sentiment API (https://twinword.com/api/) is used to fetch the sentiments of newly generated lines. The API returns the sentiment score for each word in the sentence. Hence, the sentiment score of each sentence is calculated by taking the average of the sentiment scores of its words. The obtained score can be positive, negative or neutral. If the score is below -0.05, it is tagged as negative and if it is above 0.05, then it is positive. If the score falls within this range, the sentiment of the sentence is marked as neutral.

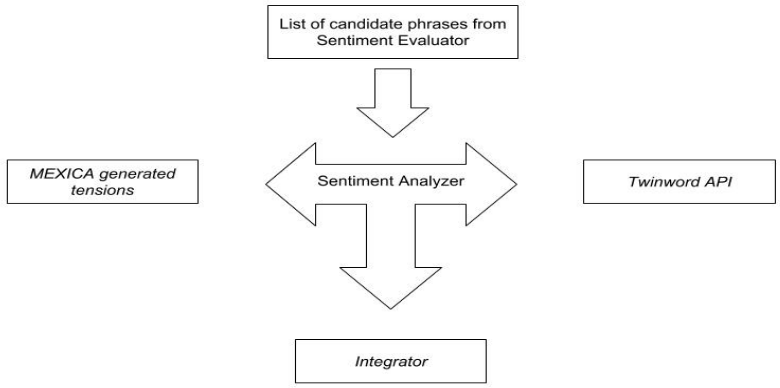

Fig. 5. Flowchart of sentiment analyzer.

The sentiments of storylines are extracted from MEXICA's tensions which gives information about the type of emotion present between the characters in the narrative line. The emotion can be love competition or clash in emotions. In either case, MABLE selects those candidate lines that contain opposite sentiments as that of in the storyline. After sentiment analysis is performed on the figurative lines and narrative sentences, the candidate lines are reduced to those that carry the similar sentiments as that of the narrative line.

3) Integrator: The point of view of the emotionally connected lines to the story need to be neutralized using the Integrator. While MEXICA tells narratives in the third person, most of the lyrics on which Markov model is trained are written in the first person. Therefore, integration requires a change from a first person to a third person point of view. Hence, the first and second person pronouns are substituted with third person pronouns in order to connect the figurative lines with the narrative sentences as shown in Fig. 6.

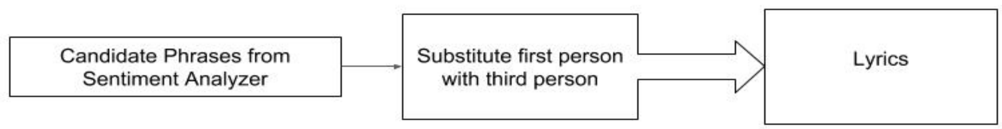

Fig. 6. Flowchart of integrator.

B. A Step-by-Step Example

The goal is to create a new line for each of the narrative line in the story and those new sentences will follow them in the lyrics. Consider a narrative line obtained from MEXICA: “The priest was ambitious”. This subsection discusses how this story line uses MABLE’s components to generate a new candidate line that rhymes with it, has a similar sentiment score, and contains the same number of syllables.

The narrative line is tokenized into a list of words - [‘The’, ‘priest’, ‘was’, ‘ambitious’]. Then, a list of words that rhyme with the last word in the sentence i.e. “ambitious” is obtained using CMUdict. In this example, the pronunciation of ambitious attained is [u'AE0', u'M', u'B', u'IH2', u'SH', u'AH0', u'S'] and the list formed is [vivacious, victorious, imperious, lustrous, ...]. The rhyme quality score is calculated by counting the number of consecutive matching pronunciation elements. If two words have same rhyme quality score, then preference is given to those words whose pronunciations end with the same vowels or the words that have the same vowels at same places, for example, soon and pool.

Then, batch Markov method is called to get candidate sentences whose last words rhyme with the above bag of words that excludes the sentences ending with exactly the same word. Some examples of candidate sentences are as follows:

Wonder how they judge us

From the city sleeps

It happens here in this world a crazy place

Left without a trace

How many nights like

this

For a thousand ships

Down in the darkness

A dream is a curse

These lines do not perfectly rhyme with the storyline. MABLE chooses low-quality rhyming lines if perfect rhymes are not found. This variety of rhyme quality makes lyrics resemble human-made lyrics where imperfect rhymes are so common.

Then, the model reduces this list by matching the number of syllables with those in MEXICA's sentence. So, the list becomes:

Left without a trace

For a thousand ships

From the city sleeps

Down in the

darkness

A dream is a curse

Sentiment analysis eliminates the sentences having different sentiments to that of MEXICA's narrative line. NLP TwinWord API is used to categorize lines into positive, negative or neutral sentiments. Hence, the list along with their sentiment scores looks like this:

Left without a trace, 0

For a thousand ships, 0

From the city sleeps, 0

Down in the darkness, -3

A dream is a curse, -1

In these lines, 0 represents positive, -1 and -3 represent negative sentiments. Here, the sentence with "-3" score has more negative sentiments than sentence having "-1" score. Since the narrative sentence is positive, other sentences having negative sentiments are eliminated. Therefore, the final set consists of the following three phrases:

Left without a trace, 0

For a thousand ships, 0

From the city sleeps, 0

A random sentence is picked from it and that line goes to Integrator where it substitutes the first person with third person pronouns.

The resulting lyrics are as follows:

The priest was ambitious

For a thousand ships

This process is repeated for every line in the story. Some examples of lyrics generated by MABLE are shown in Fig. 7 [21, Fig. 2] and Fig. 8 [21]. Every odd number sentence is created by MEXICA, whereas every even-numbered sentence is created using MABLE thus generating a suitable candidate line for each of the narrative sentences.

The priest was born under grace of the great god
And evolving from the shadows lifted
The lady was an inhabitant of the great city
But just remember there's a sign of intensity

The lady wanted him from the start
The friends that they are
The lady hid her love for the priest
They just don't think they have got a secret

But she fell in love with him
They don't know of the time
The princess was in love with the priest
Wondering where would they have got

The princess admired the lady
Movies only make them crazy
She felt much affection for her
They thought the world were on fire

The priest was ambitious
He will be out of place
He wanted power easily
As time goes by so slowly

Fig. 7. Example of lyrics made by MABLE.

But she fell in love with him
Girl when they feel the same
The princess was in love with the priest
Can't let go and it never goes out

She also abominated what he did
Be the things they said
The princess was shocked by the priest's actions
And though her heart cant take it all happens[1]

Fig. 8. Another example of lyrics created by MABLE.

VI. CONCLUSION and FUTURE WORK

This project outlines how MABLE, an automated narrative lyric writing model, conveys narratives in poetic form. The model divides its work into three main components, where the first component provides a set of candidate lines satisfying the poetry format constraints, the second aims to filter those phrases by matching their sentiments with the story, and the last one finally integrates them into a ballad.

This report demonstrates how automated poetry generation may support people in writing their own lyrics. Such tools may not only help people in their creative pursuits but also may assist them in developing their skills required for writing lyrics. Human learners could collaborate with automated systems, thus following the process of generating and evaluating the lyrics together. A similar collaboration could be seen between MABLE and users. For example, MABLE would provide the user with the first draft of lyrics. Then, the user would evaluate the first draft and modify some of its pieces and let MABLE know about the required updates in the existing one. Next, MABLE could provide some feedback to the user. Moreover, competitions could be organized in which human judges could evaluate lyrics created by MABLE and the learners.

One of the interesting characteristics of MABLE is its integration with MEXICA. Although MABLE relies heavily on its narrative writing ability, it still could be expanded in novel ways. Rather than generating a story and then modifying it, MABLE has integrated both of the processes. Thus, preserving the existing one, a new art form is created. Additionally, MABLE could be modified by automating the process of pulling out the corpus of lyrics from the web that better suits the content of the stories used for the generation of lyrics. Furthermore, it would also be interesting to combine MABLE with a melody-writing system in order to construct complete songs.

REFERENCES

[1] H. Manurung, “An evolutionary algorithm approach to poetry generation,” Ph.D. dissertation, College of Science and Engineering, University of Edinburgh, Scotland, UK,
2004. [Online]. Available:
https://www.inf.ed.ac.uk/publications/thesis/online/IP040022.pdf

[2] N. Ye, “A markov chain model of temporal behavior for anomaly detection,” in *2000 Proc. IEEE Systems, Man, and Cybernetics Information Assurance and Security Workshop,* West Point, New York, vol. 166, p. 169. [Online]. Available:
https://homepages.laas.fr/owe/METROSEC/DOC/WA1_1.pdf

[3] A. Reshamwala, D. Mishra, and P. Pawar, “Review on natural language processing,” *IRACST Engineering Science and Technology: An International Journal (ESTIJ)*, vol.3, no. 1, pp. 113-116, 2013. [Online]. Available:
https://www.researchgate.net/profile/Alpa_Reshamwala/publication/
235788362_REVIEW_ON_NATURAL_LANGUAGE_PROCESSING/links/00463516276f412048000000.pdf

[4] C. D. Manning and H. Schutze, *Foundations of Statistical Natural Language Processing*. Cambridge, UK: MIT Press, 1999. [Online]. Available:
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.121.2604&rep=rep1&type=pdf

[5] K. Monteith, T. Martinez, and D. Ventura, “Automatic generation of melodic accompaniments for lyrics,” in 2012 *Proc. Int. Conf. on Computational Creativity*, pp. 87-
94. [Online]. Available: http://computationalcreativity.net/iccc2012/wpcontent/uploads/2012/05/087-Monteith.pdf

[6] E. Charniak, “Statistical parsing with a context-free grammar and word statistics,” *AAAI/IAAI*, vol. 2005, no. 598-603, p. 18, 1997.

[7] P. Goncalves et al., “Comparing and combining sentiment analysis methods,” in *Proc. 1st ACM Conf. on Online Social Networks*, pp. 27-38, 2013. [Online]. Available:
https://arxiv.org/pdf/1406.0032.pdf

[8] K. V. Deemter, M. Theune, and E. Krahmer, “Real versus template-based natural language generation: A false opposition?” *Computational Linguistics*, vol. 31, no. 1, pp.
15-24, 2005. [Online]. Available:
http://www.mitpressjournals.org/doi/pdf/10.1162/0891201053630291

[9] H. Nguyen and B. Sa, “Rap lyric generator,” New York, USA. pp. 1-3, 2009. [Online]. Available:
https://pdfs.semanticscholar.org/7d67/2ca1566b5573dd350bc65ab3c26758cbef62.pdf

[10] E. Malmi, P. Takala, H. Toivonen, T. Raiko, and A. Gionis, "Dopelearning: A computational approach to rap lyrics generation," 2015. [Online]. Available: https://arxiv.org/pdf/1505.04771.pdf

[11] H. G. Oliveira, “Tra-la-lyrics 2.0: Automatic generation of song lyrics on a semantic domain,” *Journal of Artificial General Intelligence*, vol. 6, no. 1, pp. 87-110, Dec. 2015. [Online]. Available: https://www.degruyter.com/downloadpdf

[12] S. Colton, J. Goodwin, and T. Veale, "Full face poetry generation," in *2012 Proc. 3rd Computational Creativity Int. Conf.*, pp. 95-102. [Online]. Available:
http://computationalcreativity.net/iccc2012/wpcontent/uploads/2012/05/095Colton.pdf

[13] S. Pudaruth, B. F. Bhaukaurally, and M. H. A. Didorally, “A semi-automated lyrics generation tool for mauritian sega,” *IAES Int. Journal of Artificial Intelligence*, vol. 1, no.
4, pp. 201-213, Dec. 2012. [Online]. Available: https://search.proquest.com/openview/5b6838c51f8a1a506bcab26a317f11db/1?pq-origsite=gscholar&cbl=1686339

[14] B. C. L. Tobing and R. Manurung, “A chart generation system for topical metrical poetry,” in *2015 Proc. 6th Int. Conf. on Computational Creativity*, pp. 308-314. [Online]. Available: http://axon.cs.byu.edu/ICCC2015proceedings/14.2Tobing.pdf

[15] J. Misztal and B. Indurkhya, “Poetry generation system with an emotional personality,” in *2014 Proc. 5th Int. Conf. on Computational Creativity*, pp. 72-81. [Online]. Available: https://pdfs.semanticscholar.org/d89d/053b1c2481088b1af2bd36e0a6d959ff1373.pdf

[16] M. Scirea, A. B. Barros, N. Shaker, and J. Togelius, “SMUG: Scientific music generator,” in *2015 Proc. 6th Int. Conf. on Computational Creativity*, p. 204. [Online]. Available: http://axon.cs.byu.edu/ICCC2015proceedings/9.2Scirea.pdf

[17] J. M. Toivanen, M. Järvisalo, and H. Toivonen, “Harnessing constraint programming for poetry composition,” in *2013 4th Int. Conf. on Computational Creativity*. [Online]. Available:
https://helda.helsinki.fi/bitstream/handle/10138/42894/toivanen_constraints_poetry_icc
c
2013.pdf?sequence=2

[18] M. Ghazvininejad, X. Shi, Y. Choi, and K. Knight, "Generating topical poetry," in *2016 Proc. Natural Language Processing Empirical Methods Conf.*, pp. 1183-1191. [Online]. Available: https://www.aclweb.org/anthology/D/D16/D161126.pdf

[19] R. Pérez y Pérez, “A computer-based model for collaborative narrative generation,” *Cognitive Systems Research*, vol. 36, pp. 30-48, Dec. 2015.

[20] R. Pérez y Pérez, “Employing emotions to drive plot generation in a computer-based storyteller,” *Cognitive Systems Research*, vol. 8, no. 2, pp. 89-109, Jun. 2007. [Online]. Available:
https://pdfs.semanticscholar.org/759d/4bc5e311d4730a6386e584adcdf2980ef2c5.pdf

[21] D. Singh, M. Ackerman, and R. Pérez y Pérez, “A Ballad of the Mexicas: Automated lyrical narrative writing,” in *8th Int. Conf. on Computational Creativity, ICCC,* Atlanta, 2017. [Online]. Available:
https://pdfs.semanticscholar.org/69f0/a24d62d92632fd726d455fbf6e9053d67b2a.pdf