Coherent Lyrics Generation

Lisa Fan **David Lowell** **Mengyuan Wang**
Khoury College of Computer Science
Northeastern University
Boston, MA
{fan.lis, lowell.d, wang.mengyua}@husky.neu.edu
UIDs: 001177470 001881166 001493091

1 Introduction

Lyrics serve an important role in music. The use of natural language allows the explicit expression of emotions, themes, and events. But because lyrics employ text in conjunction with melody, they are constrained with respect to rhythm in ways that prose text is not. Lyrics are further constrained by artistic conventions, including rhyming schemes as well as verse and chorus structure. In order to adhere to these restrictions, lyrics often use sentence fragments or highly figurative language and don't strictly follow grammar rules.

Prior work on computer-assisted lyrics generation has mainly focused on satisfying the former constraints of rhythm and rhyme, overlooking the importance of semantics in the produced lyrics (Barbieri et al., 2012; Potash et al., 2015). Recently, Watanabe et al. (2018) used a neural network to improve the discourse structure of generated lyrics conditioned on melody. Consider the following example verse generated by their model, translated from Japanese using Google Translate[1]: "I just want to believe in love / **Let me tell you my tears /** Do not think about anything / I wish I could be honest." Because neural language models naturally excel at generating grammatically correct output, this proposed model produces syntactically correct lyrics. However, these lyrics often do not make much semantic sense (see the line in **bold** above). Furthermore, while typical song verses stay on a single theme, there is little cohesion between the lines in this verse.

We address these challenges in this work by proposing a hierarchical model for semantically coherent lyric generation[2] To improve the coherence and cohesion of lyrics generated by the current state-of-the-art model (Watanabe et al., 2018), we introduce an intermediate step that learns to generate semantic representations of the lyrics. The structure is similar to the two step text generation process of context selection (ie. "what to say") and surface realization (ie. "how to say it") (Angeli et al., 2010; Konstas and Lapata, 2012; Mei et al., 2015), which is intended to mirror the human song-writing process. A human author begins with a semantic concept, informed by their own stylistic and artistic preferences. This concept is then used to create lyric lines which tell a cohesive story or stay on a similar theme. Because songwriting styles are highly dependent on the artist, we also experiment with explicitly incorporate artist information as an input feature to our model.

In order to understand how the various components of our proposed hierarchical architecture affects model performance, we conducted an extensive ablation study. Our results showed that although our proposed model design may not be sufficient in fully capturing semantic information, our semantic models successfully outperformed their counterpart baselines with respect to generating semantically meaningful and consistent lyrics. A pilot human evaluation study, however, showed that the semantic models made sacrifices in terms of other metrics of quality, such as grammaticality.

[1]The validity of this translation was confirmed by one of our members (Lisa) who speaks fluent Japanese.
[2]Our code can be found at: `https://github.com/lisafan/lyrics_generation`

1.1 Contributions

Our contributions are as follows:

- Designed a hierarchical recurrent neural network (RNN) architecture to enhance semantic coherence in the task of lyrics generation.
- Made the model conditional to produce lyrics that are similar in style to a user-specified artist and align with user-specified melody.
- Demonstrated both qualitatively and quantitatively the increased semantic coherence of lyrics produced by our hierarchical RNN as compared to non-hierarchical baselines through an extensive ablation study.

2 Related Work

Lyrics generation can be considered a specific application of poetry generation, which is a well studied problem (Zhang and Lapata, 2014; Oliveira, 2012; Colton et al., 2012; Tosa et al., 2008). Though lyrics generation and lyric-conditioned melody generation have both been researched (Ramakrishnan A et al., 2009; Fukayama et al., 2012), we have only found three studies on melody-conditioned lyric generation. Two endeavors into this task employ a heuristic (Gonçalo Oliveira et al., 2007) or statistical (Ramakrishnan A et al., 2009) method and produce only short lines at a time.

Recently, Watanabe et al. (2018) proposed a successful RNN language model that can produce Japanese lyrics for an entire song. Their model uses a sliding window of a portion of the melody (a sequence of notes represented by a duration and pitch) as input and outputs a word and the word's syllable count at each timestep. The syllable count is used to center the sliding window for the next timestep. One major challenge not faced in their work stems from the fact that Japanese syllable-letter structure is significantly different from English. A single Japanese *hiragana* character always corresponds to a single syllable, whereas determining syllable counts for English words is more complex. This makes the task of lyrics-melody alignment even more difficult for the English language.

Our work can also be considered a form of author conditioned text generation. Tikhonov and Yamshchikov (2018) explore author conditioned poetry generation, a domain closely related to lyric generation. Their work augments an LSTM by providing as additional input a concatenation of document and author embedding at each time step. A related area of research, as explored by Ficler and Goldberg (2017), uses a similar method to condition generated text on style features such as whether the output should sound professional or personal rather than directly on author.

In designing our architecture, we draw on hierarchical RNNs (Sordoni et al., 2015; Serban et al., 2016; Chung et al., 2016) that learn to first generate a coarse-grain representation of the output, before using another network to generate the ultimate fine-grained output. Li et al. (2015) demonstrates a hierarchical structure that uses one RNN to output a sentence-level embedding, and another RNN that uses the sentence embedding to generate words sequentially. Hierarchical models can allow us to learn an underlying semantic representation, from which we generate a textual representation that is reflective of those semantics. This is similar in principle to MoCoGAN (Tulyakov et al., 2018), in which video is generated from underlying content and motion representations that are generated first. We apply this to lyric generation by generating song semantics from which we generate lyrics.

3 Method

Our proposed hierarchical lyrics generation model uses two RNNs to learn lyrics semantics and lyrics-melody alignment respectively (See Figure 1). The first RNN learns to produce one semantic representation vector per lyrics line in the final generation. The second RNN uses that semantic representation and a melody context vector to produce one line of lyrics. One advantage to our hierarchical framework is that the lyric lines can be generated in parallel, thus reducing training time.

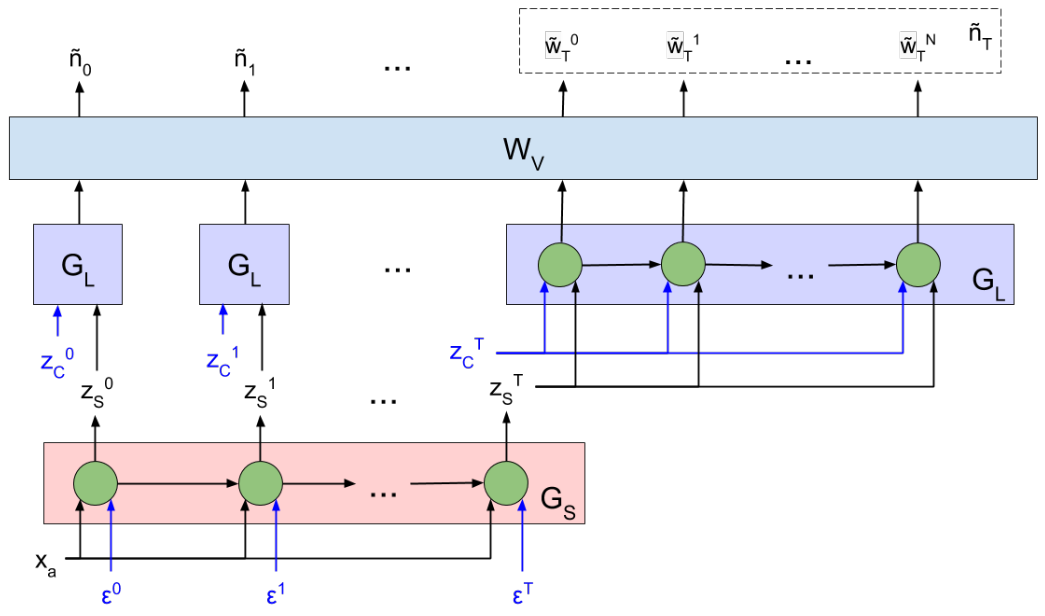

Figure 1: Our proposed model. The semantic generator RNN, $\mathbf{G_S}$, will take in a noise vector, ϵ^t, and optionally an artist vector, $\mathbf{x_a}$, at each timestep t and generate a semantic representation vector, $\mathbf{z}_s^t$, for one lyric line. The lyrics generator RNN, $\mathbf{G_L}$, will take in the semantic representation, $\mathbf{z}_s^t$, and the melody context vector, $\mathbf{z}_c^t$, of the song line at timestep i. This output passed through the linear transformation W_v produces a sequence of words, $[\widetilde{w}_t^0, ..., \widetilde{w}_t^N]$. Best viewed in color.

Let a line of generated lyrics, or a sequence of generated words $\widetilde{w}$, be represented by $\widetilde{n}$, and the sequence of generated lines for the entire song be represented by $\widetilde{m}$.

$$\widetilde{n}_t = [\widetilde{w}_t^0, ..., \widetilde{w}_t^N] \tag{1}$$

$$\widetilde{m} = [\widetilde{n}_0, ..., \widetilde{n}_T] \tag{2}$$

Our model is comprised of two separate long short-term memory (LSTM) units (Hochreiter and Schmidhuber, 1997). The first LSTM, $\mathbf{G_S}$, is a semantic generator that takes as input a random noise vector sampled from a Gaussian distribution, ϵ^t, and an optional artist vector, $\mathbf{x}_a$, at each timestep t. The artist information can be left out entirely (Eq. 3a), or be encoded as a learnable embedding (Eq. 3b). It outputs a semantic representation vector, $\mathbf{z}_s^t$, for one line in the final lyrics:

$$\mathbf{z}_s^t = \mathbf{G_S}\left(\left[\mathbf{z}_s^{t-1} \,\|\, \epsilon^t\right]\right) \tag{3a}$$

$$\mathbf{z}_s^t = \mathbf{G_S}\left(\left[\mathbf{z}_s^{t-1} \,\|\, \epsilon^t \,\|\, \mathbf{x}_a\right]\right) \tag{3b}$$

One pass through the lyrics generator, $\mathbf{G_L}$, produces a single line of lyrics. This second LSTM takes as input the semantic representation vector from the semantic generator as well as a melody context vector for the line, $\mathbf{z}_c^t$, and produces output $\mathbf{y}_t^i$ at each timestep i. The output undergoes a linear transformation and a softmax function to get a vocabulary distribution for the generated word, $\widetilde{w}_t^i$.

$$\mathbf{y}_t^i = \mathbf{G_L}\left(\left[\mathbf{y}_t^{i-1} \,\|\, \mathbf{z}_s^t \,\|\, \mathbf{z}_c^t\right]\right) \tag{4}$$

$$\mathbf{o}_t^i = \mathrm{W}_v \mathbf{y}_t^i + \mathbf{b} \tag{5}$$

$$p\left(\widetilde{w}_t^i\right) = \mathrm{softmax}\left(\mathbf{o}_t^i\right) \tag{6}$$

The lyrics generator is applied to each line of the lyrics in parallel. We represent the melody context vector as a sequence of frequencies taken from the input melody at regular time intervals (every 0.2 seconds).

4 Experiment

4.1 Data

We train our model using data derived from the DALI corpus (Meseguer-Brocal et al., 2018)[3], which is a karaoke based collection of annotated audio tracks, primarily composed of rock and pop songs. For each song, the corpus provides performance audio, melody data, and aligned text lyrics. Lyrics are aligned hierarchically, with annotations available at a paragraph, line, word and syllable level. At each level, text is time aligned with start and end time stamps. Melody data is available as frequencies with one note per syllable.

We use the English language subset of this data, which is comprised of 3,953 songs by 1,462 artists. The songs have an average of 53.9 lines per song and 5.3 words per line. We use 90% of the data for training and the other 10% as a held out test set. In order to further increase our dataset size, we split each song's lyrics into all possible chunks of 5 consecutive lines, giving us 177,510 training samples.

For models that condition on artist, we require a significant number of training samples per artist. Since the DALI dataset only contains a few songs from each artist, we pretrain our semantic generator with a lyrics dataset from Kaggle[4]. We chose songs from a subset of 42 common artists in the two datasets (see Appendix A for details). Each artist has at least 10 songs from DALI and 100 songs from Kaggle. This subset has 10,674 songs, for which 457 have melody data. The songs have an average of 33.2 lines per song and 7.2 words per line.

4.2 Training Details

Due to the lack of baselines applicable to our task, we focus on an ablation study using various steps in our model. We perform evaluation on the following models:

- **lyrics-only:** The simplest baseline. The model is not conditioned on the artist and the second LSTM, $\mathbf{G_L}$ is absent. Instead, the output from $\mathbf{G_S}\left(\left[\mathbf{z}_s^{t-1} \parallel \epsilon^t\right]\right)$ is fed directly into the linear layer: $\mathbf{y}_t^i = \mathbf{z}_s^t$.
- **with-semantics:** A hierarchical model with the melody context vector, $\mathbf{z}_c^t$, omitted from the input: $\mathbf{G_L}\left(\left[\mathbf{y}_t^{i-1} \parallel \mathbf{z}_s^t\right]\right)$.
- **no-semantics:** A baseline for the hierarchical model, where $\mathbf{G_S}$ and the melody context vector are not used and $\mathbf{G_L}$ is given random noise as input: $\mathbf{z}_s^t = \mathcal{N}(0, 1)$.
- **melody-only:** An LSTM that takes in a melody context vector as input and generates aligned lyrics: $\mathbf{G_L}\left(\left[\mathbf{y}_t^{i-1} \parallel \mathbf{z}_c^t\right]\right)$. $\mathbf{G_S}$ is not used.
- **semantic-melody:** The full model. A hierarchical model where $\mathbf{G_S}$ generates semantic representations, and $\mathbf{G_L}$ generates lyrics based on the semantic representation and a melody context vector.

We also trained variations of the models using conditional input artists, but found that these models performed poorly. Details and discussion of the conditional artist models can be found in Section 4.4. The rest of this paper will focus on models that did not condition on an input artist.

All models were trained on the same vocabulary of size 6,434, where we included all words that appeared in our training data at least 5 times. A special unknown token was used to replace all words outside the vocabulary. Lyrics were lower cased and tokenized using NLTK's word tokenizer. The maximum output size was 5 lines of 20 tokens each.

Our models used 2-layer LSTMs with 64-dimensional hidden states and 128-dimensional word embeddings. Melody conditioned models use melody context vectors of size 30. All models were trained with a batch size of 16, using the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 0.005.

[3] `https://github.com/gabolsgabs/DALI`

[4] `https://www.kaggle.com/gyani95/380000-lyrics-from-metrolyrics`

Model	Perplexity
lyrics-only	72.290
lyrics-only + artist	76.673
semantic-melody + artist	276.334

Table 1: Results for artist-conditioned models. Perplexity is calculated on a held-out validation set. Lower is better. "+ artist" indicates models using artist information as conditional input.

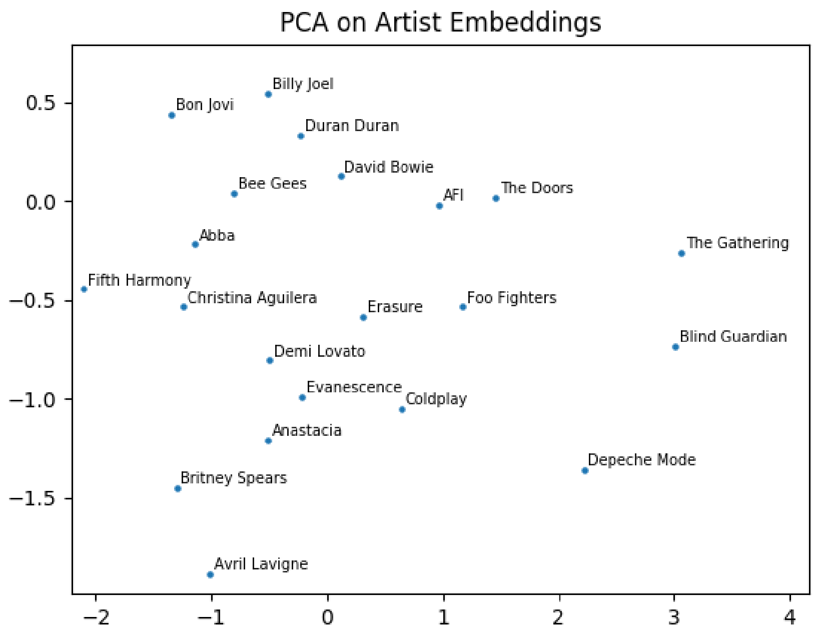

Figure 2: Principal Component Analysis (PCA) on select artist embeddings from the lyrics-only + artist model. Some artists are omitted to improve visibility.

4.3 Evaluation Metrics

We use the standard test-set perplexity (PPL) measure as an automatic evaluation metric. PPL measures the predictability of wording in the original lyrics, where a lower PPL value indicates that the model can generate fluent lyrics.

We also conducted a pilot human evaluation study on 10 random samples from each of the 5 model variations described in Section 4.2. Following the procedure of Watanabe et al. (2018), participants were asked to rate the generated lyrics on four different categories using a five-point Likert scale. The four categories include **grammaticality** (whether the lyrics are grammatically correct), **line-level meaning** (whether individual lyric lines make semantic sense), **document-level meaning** (whether the entire generated lyrics stay on a similar theme), and **overall quality**. The study was completed by 3 participants.

4.4 Conditioning on Artist

Early in our research, we observed that different artists produce lyrics that are highly varied both semantically and syntactically. We felt that training on lyrics from a variety of artists would confuse the model, and therefore chose to incorporate artist information as an explicit input feature.

We found that although the artist conditioned lyrics-only model only performed slightly worse than the model not conditioned on artist, the melody- and artist-conditioned model performed significantly

lyric-only:	We could be the man
	I don’t wanna be in the world
	We’re gonna run back to me
	Try to do what I need to
	Fine and what they’re gone
no-semantics:	That I’m breaking good
	You didn’t runaway
	Too evil we say I could fight and here
	That I’m in your lights and believe you’re breathing
	Will think to be a going on
with-semantics:	I’ve had to pray lay down the same body rise in the picture I can’t <UNK> <UNK> we
	<UNK> and you see it ’s you <UNK> there won’t I’ll have ever there is alive and my
	Kick on our heart and i won’t <UNK> <UNK> <UNK> and I’m livin’ the world go then
	This is the day I say i don’t shine now something
	It ’s broken wild and they’ve never we don’t <UNK> and you see you will kill me they

Figure 3: Sample outputs from our unconditioned models. See Section 4.2 for details on each model.

melody-only:	Have me’s fantasy (5)	My love, it lies so deep (6)
	This I can always keep my reason (9)	Ever dream of me (5)
	And I never got the life to love out (10)	Ever felt away with me (7)
	so I feel the little man (7),	Just once that all I need (6)
	I need you change to her (6)	Entwined in finding you one day (8)
semantic-melody:	Or I want them, (4)	You know I love you (5)
	Oh I’ll make oh I’m love (6)	I always will (4)
	You’m all be try to I want with me (9)	My mind’s made up (4)
	Some your down love (4)	By the way that I feel (6)
	Till oh a and you’ll you got all seems da, that gone in <UNK> and when not I (17)	There’s no beginnin’ (5)

Figure 4: Sample outputs from our melody conditioned models (left) with the original, human-written lyrics associated with the input melody (right). Numbers in parentheses are per line syllable counts. See Section 4.2 for details on each model.

worse than both models (See Table 1). We attribute this to the fact that we did not have enough training data per artist. As described in Section 4.1, the Kaggle/DALI subset data contained less than 500 songs with melody information, meaning each of the 42 artists only had an average of around 10 songs. As a result, these models are severely underfitted.

Despite the poor performance of these models, a principle component analysis (PCA) of the learned artist embeddings showed that artist-conditioned models are successful in identifying certain artist styles. Figure 2 shows that artists with similar styles are successfully clustered together: female pop singers are clustered in the bottom left; The Gathering (a Dutch rock band) and Blind Guardian (a German power metal band) are close together on the right. This suggests that given a larger training data, artist embeddings may help in the task of lyrics generation.

Model	Perplexity	Avg Line Len
lyrics-only	62.295	6.059
no-semantics	66.170	9.124
with-semantics	65.348	17.964
melody-only	91.761	5.590
semantic-melody	91.294	8.377

Table 2: Results on our 5 model variations. Perplexity is calculated on a held-out validation set (Lower is better). Avg Line Len is the average number of tokens per line for 50 generated samples.

5 Results

5.1 Qualitative Results

We present sample output verses from our five models in Figures 3 (non-melody conditioned) and 4 (melody conditioned). The lyric-only and no-semantic verses demonstrate the lack of coherence that we hope to address. Lines from these models generally obey the grammatical structure of English. However, individual lines, such as “Fine and what they’re gone” or “Will think to be a going on”, often lack a coherent meaning. In cases where individual lines do have some meaning, that meaning is not related to other lines in the verse. For example, the lines “we could be the man”, “I don’t wanna be in the world”, and “Try to do what I need to” each represent a coherent notion, but they are unlinked and disjointed.

In contrast, the lyrics generated by the with-semantic model are less grammatically sound, but exhibit a greater thematic consistency across lines. Specifically, in this example we see repeated use of phrases related to life and death. "pray lay down the same body rise", “there is alive”, “kick on our heart”, “I’m livin’ ”, and “you will kill me” all reflect this theme. Where the model seems to struggle is in linking these phrases in a grammatical manner.

In the melody conditioned domain we observe similar results, with the semantic-melody model producing lyrics which are more meaningful than the melody-only model. Here we observe that the semantic-melody lyrics stay on the general theme of love, which is also the the theme of the original, human-written lyrics associated with the input melody. We also observe that in both models, the syllable counts of generated lines are not related to the syllable counts of the original, human-written lyrics for the input melodies. This suggests that the model is not learning the relationship between syllable and note structure.

5.2 Quantitative Results

Table 2 shows the results of calculating perplexity and the average number of tokens per line on our models. The with-semantics model outperforms the no-semantics model, which feeds the lyrics generator noise instead of the learned semantic representations. Similarly, the semantic-melody model outperforms the melody-only model, thus showing that the semantic generator does indeed improve on the quality of generated lyrics. We believe we could further improve our semantic generator by explicitly modeling semantic information. In future work, we would like to experiment with using features such as tf-idf to identify key content words in the target lyrics.

The results on perplexity also shows that the lyrics-only baseline model outperforms our other models. We attribute this to the fact that a sentence can span more than a single lyrics line, an oversight that led to a flaw when designing our model architecture. We intended the semantic generator to model the general theme along the lyric lines and the lyrics generator to model the grammatical realization of that theme. By missing that crucial observation, our model architecture no longer followed that constraint.

The perplexity table also shows that our melody-conditioned models perform significantly worse than the other models. We believe this is due to the fact that our melody-conditioned models do not successfully learn the association between the melody vector and generated words. This is illustrated in figure 4, in which the model-generated and human-composed lyrics for the same melody show

Models	Grammaticality	Line Meaning	Whole Meaning	Overall Quality
lyrics-only	3.13 ± 1.04	2.77 ± 0.90	2.37 ± 0.76	2.63 ± 1.00
no-semantics	2.20 ± 0.71	2.03 ± 0.61	1.90 ± 0.76	1.97 ± 0.72
with-semantics	2.23 ± 0.73	1.97 ± 0.72	2.00 ± 0.79	2.00 ± 0.79
melody-only	2.63 ± 0.72	2.43 ± 0.73	1.90 ± 0.76	2.17 ± 0.79
semantic-melody	1.70 ± 0.60	1.80 ± 0.61	1.73 ± 0.64	1.77 ± 0.63

Table 3: Results for human evaluation. The lyrics-only model outperforms the other models across categories.

significantly different syllable counts. Therefore in future work, we would model lyric to melody alignment by explicitly incorporating syllable count as is done in Watanabe et al. (2018). Since the DALI dataset contains alignments at the line, word, and syllable level, such a task would be quite feasible.

As mentioned in Section 4.1, the songs in our DALI English subset had an average of 5.3 tokens per line. Our semantic models generate longer lyric lines than their non-semantic counterparts. The with-semantic model in particular generates exceptionally long lyric lines (See also the example in Figure 3).

5.3 Human Evaluation

We evaluate our results by surveying three volunteers who evaluate randomly chosen and selected verses, as described in Section 4.3. The results of this survey are shown in Table 3. As with the perplexity measure, human evaluation shows the lyrics-only model to outperform the others.

Our aim in this paper has been specifically to improve the semantic quality of lyrics. In our survey, this is represented by the whole verse evaluation. Of the hierarchical models, semantic-only performs the best for this metric. However, we note that this comes at the expense of lower standing with respect to the other metrics. We suggest that this may be a reflection of the fact that the semantic models produce significantly longer lines than the baselines, as shown in Table 2, thereby having more opportunities to introduce grammatical errors and extraneous words. Intuitively, it is easier for a human to disregard a single erroneous word in a four sentence utterance than it is to disregard four erroneous words in a sixteen word utterance.

6 Conclusion

In this paper, we presented a novel hierarchical neural model for semantically coherent lyrics generation conditioned on melody and artist. The model used a two step process in which the first RNN learns to generate a cohesive sequence of semantic representations and a second RNN learns to generate lyrics to fit a given melody. We also conducted an extensive ablation study to better understand how the model performance is affected by using our proposed architecture. Our ablation study showed that our semantic models produce outputs which may be less grammatically sound than those produced by baseline models, but which outperform them with respect to the semantic structure of true lyrics. We did observe several shortcomings of our model, including a failure to learn syllable-melody relations, a tendency to produce overly long lyric lines, and poor ability to generate multi-line sentences. These issues could be rectified in future work by explicitly modeling semantic representations, incorporating syllable level input, and restructuring our hierarchical architecture to operate on a sentence-level rather than a lyric line level.

References

Gabor Angeli, Percy Liang, and Dan Klein. 2010. A simple domain-independent probabilistic approach to generation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 502–512. Association for Computational Linguistics.

Gabriele Barbieri, François Pachet, Pierre Roy, and Mirko Degli Esposti. 2012. Markov constraints for generating lyrics with style. In *Ecai*, volume 242, pages 115–120.

Junyoung Chung, Sungjin Ahn, and Yoshua Bengio. 2016. Hierarchical multiscale recurrent neural networks.

Simon Colton, Jacob Goodwin, and Tony Veale. 2012. Full-face poetry generation. In *ICCC*, pages 95–102.

Jessica Ficler and Yoav Goldberg. 2017. Controlling linguistic style aspects in neural language generation. *arXiv preprint arXiv:1707.02633*.

Satoru Fukayama, Daisuke Saito, and Shigeki Sagayama. 2012. Assistance for novice users on creating songs from japanese lyrics. In *ICMC*.

Hugo R Gonçalo Oliveira, F Amilcar Cardoso, and Francisco C Pereira. 2007. Tra-la-lyrics: An approach to generate text based on rhythm. In *Proceedings of the 4th. International Joint Workshop on Computational Creativity*. A. Cardoso and G. Wiggins.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ioannis Konstas and Mirella Lapata. 2012. Concept-to-text generation via discriminative reranking. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 369–378. Association for Computational Linguistics.

Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. 2015. A hierarchical neural autoencoder for paragraphs and documents. *arXiv preprint arXiv:1506.01057*.

Hongyuan Mei, Mohit Bansal, and Matthew R Walter. 2015. What to talk about and how? selective generation using lstms with coarse-to-fine alignment. *arXiv preprint arXiv:1509.00838*.

Gabriel Meseguer-Brocal, Alice Cohen-Hadria, and Geoffroy Peeters. 2018. Dali: a large dataset of synchronized audio, lyrics and notes, automatically created using teacher-student machine learning paradigm. In *19th International Society for Music Information Retrieval Conference*.

Hugo Gonçalo Oliveira. 2012. Poetryme: a versatile platform for poetry generation. *Computational Creativity, Concept Invention, and General Intelligence*, 1:21.

Peter Potash, Alexey Romanov, and Anna Rumshisky. 2015. Ghostwriter: Using an lstm for automatic rap lyric generation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1919–1924.

Ananth Ramakrishnan A, Sankar Kuppan, and Sobha Lalitha Devi. 2009. Automatic generation of tamil lyrics for melodies. In *Proceedings of the Workshop on Computational Approaches to Linguistic Creativity*, pages 40–46. Association for Computational Linguistics.

Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. 2015. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 553–562. ACM.

Alexey Tikhonov and Ivan P. Yamshchikov. 2018. Guess who? multilingual approach for the automated generation of author-stylized poetry. *2018 IEEE Spoken Language Technology Workshop (SLT)*.

Naoko Tosa, Hideto Obara, and Michihiko Minoh. 2008. Hitch haiku: An interactive supporting system for composing haiku poem. In *International Conference on Entertainment Computing*, pages 209–216. Springer.

Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. 2018. Mocogan: Decomposing motion and content for video generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1526–1535.

Kento Watanabe, Yuichiroh Matsubayashi, Satoru Fukayama, Masataka Goto, Kentaro Inui, and Tomoyasu Nakano. 2018. A melody-conditioned lyrics language model. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 163–172.

Xingxing Zhang and Mirella Lapata. 2014. Chinese poetry generation with recurrent neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 670–680.

Appendix

A Artists

We used songs from the following 42 artists:

ABBA	Bruce Springsteen	Elvis Presley
Aerosmith	Bryan Adams	Erasure
AFI	Christina Aguilera	Eurythmics
Alanis Morissette	Coldplay	Evanescence
Alphaville	The Corrs	Fall Out Boy
Anastacia	David Bowie	Fifth Harmony
Avenged Sevenfold	Demi Lovato	Fleetwood Mac
Avril Lavigne	Depeche Mode	Foo Fighters
Bad Religion	Disturbed	Frank Sinatra
Bee Gees	The Doors	Garbage
Billy Joel	Dropkick Murphys	Garth Brooks
Blind Guardian	Duran Duran	The Gathering
Bon Jovi	Electric Light Orchestra	George Michael
Britney Spears	Elton John	Girls Aloud

Artists were selected based on the number of lyrics they had in the DALI and Kaggle datasets. Our dataset contained at least 10 songs from DALI and 100 songs from Kaggle for each of the selected artists.